# Olimpiada FDI:

**Una suite de juegos educativos**

**con ranking y medallero de la FDI**

**Javier Gárate Arana**
**Alejandro González Pérez**
**Miguel Mejía Fernández**
**Alberto Rodríguez Villalobos**

**GRADO EN INGENIERÍA INFORMÁTICA**

UNIVERSIDAD COMPLUTENSE DE MADRID


**TRABAJO DE FIN DE GRADO**

**Madrid, Junio de 2015**

**Director: Pablo Moreno Ger**

# Autorización de difusión y utilización-

Javier Gárate Arana, Alejandro González Pérez, Miguel Mejía Fernández y Alberto Rodríguez Villalobos, autores del presente documento y del proyecto “Olimpiada FDI” autorizan a la Universidad Complutense a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la propia memoria, como el código, la documentación y/o el prototipo desarrollado.

**Javier Gárate Arana** **Alejandro González Pérez**

**Miguel Mejía Fernández** **Alberto Rodríguez Villalobos**

**Madrid, Junio de 2015**

# Agradecimientos

Queremos dar nuestro más sincero agradecimiento a nuestra familia, por toda la paciencia y el apoyo recibido a lo largo de estos meses y de nuestra vida universitaria.

También a nuestro director de proyecto, Pablo Moreno Ger, por todo el tiempo invertido en nosotros y su paciencia en los momentos duros.

Sin ninguno de vosotros no habría sido posible llegar hasta aquí.

A Marta por perder un poco de su valioso tiempo para echarnos una mano.

Mención a Ignacio y Luis por su pequeño granito de arena aportado.

A los participantes en el caso de estudio, por su tiempo dedicado y sus consejos para la mejora en la interacción de la plataforma.

**A todos ellos, muchas gracias.**

# Índice

Índice de Figuras

Índice de Tablas

# Índice de abreviaturas

**HTML:** Hyper Text Markup Language

**JSON:** JavaScript Object Notation

**XML:** eXtensible Markup Language (lenguaje de marcas extensible)

**URI:** Uniform Resource Identifier

**REST:** Representational State Transfer

**AS:** Application Server

**HTTP:** HyperText Transfer Protocol

**CSS:** Cascading Style Sheets

**SO:** Sistema Operativo

**API:** Application Programming Interface

**SDK:** Software Development Kit

**JVM:** Java Virtual Machine

**JDK:** Java Development Kit

**CEO:** Chief Executive Office

**MIDP**: Mobile Information Device Profile

**WP:** Windows Phone

**VAC**: Valve Anti-Cheat

**XP:** Experiencia

**SSH:** Secure Shell

**AWT:** Abstract Window Toolkit

# Resumen

El proyecto OlimpiadaFDI ha consistido en el desarrollo de una plataforma gamificada de juegos educativos, centrada en el ámbito de la Ingeniería Informática. La visión del proyecto es la creación de una plataforma inspirada en Steam o Xbox Live en la que distintos desarrolladores pueden contribuir distintos juegos, pero las puntuaciones, insignias y logros conseguidos dentro de cada juego se asocian al perfil de cada usuario.

Además, la plataforma está concebida para soportar juegos de distintas plataformas (e.g. HTML5, Android, C++, Java) que podrían ser implementadas por distintos equipos de desarrollo.

De cara a conseguir estos objetivos, el desarrollado ha consistido en la realización de la infraestructura de una plataforma de distribución de juegos educativos y servicios, tanto para los juegos como para los usuarios. Esta plataforma se basa en una página web que nos permite realizar la gestión de los usuarios, además de gestionar las puntuaciones y logros de cada usuario, la creación de grupos de usuario y la consulta de rankings de puntuación. Para completar se ha desarrollado un amplio conjunto de juegos a modo de implementaciones de referencia, mostrando como distintos juegos de distintas plataformas podrían coexistir dentro del ecosistema de juegos propuesto por OlimpiadaFDI.

Tanto los juegos como el portal web se comunican con un backend de servicios REST que lleva la gestión de usuarios, insignias, grupos y puntuaciones, dando lugar a un sistema modular y extensible.

Esto supone además que la plataforma quedará abierta para que otros desarrolladores puedan aprovecharse de ella e incluir sus propios juegos.

## Palabras clave:

Videojuegos educativos; gamificación; plataformas sociales; insignias; servicios REST.

# Summary

The project aims at developing an infrastructure of a platform for the distribution of educational games and services, both for game developers and users. This platform is supported by a website that allows us to manage the users together with the rankings, coming from the scores achieved through the “badges” of the games. At the same time it possesses the function to create groups that carry out competitions between the scores of the users from each group. To complete a wide suite of games on different platforms we´re developing the first launch of the system.

Our platform is characterized by being innovative in the field of education, offering the possibility to learn and compete at the same time; which is an incentive during the learning process. We can differentiate three main elements in the platform:

- Backend services divided into two parts, to support both games as well as users and group’s features.
- Web presentation with “badges”, users and groups management and presentation of rankings
- Suite of games in different platforms.

The platform will remain open so that other developers can take advantage of it and include their own games.

Key words

Backend, frontend, badge, user, group, platform, multiplayer, services.

# Capítulo 1. Introducción

A medida que nuestra cultura ha evolucionado hacia lo que hoy en día denominamos Sociedad del Conocimiento, la presión por encontrar nuevos modelos educativos se ha incrementado significativamente. Esto ha hecho que el clásico método de enseñanza formal empiece a quedar eclipsado por las posibilidades que traen consigo las nuevas tecnologías.

Dentro de estos modelos educativos basados en nuevas tecnologías, nos llama especialmente la atención la posibilidad de usar videojuegos como recursos de aprendizaje, en lo que distintos autores del campo denominan aprendizaje basado en juegos (Prensky, 2004) o juegos serios (Michael y Chen, 2006).

> Debemos descubrir la pasión de cada estudiante.
>
> Los profesores de hoy deberían de eliminar las clases magistrales.
>
> *Marc Prensky* (Junio y Noviembre 2014)

En realidad, esta idea de usar videojuegos como método educativo ha existido prácticamente desde el nacimiento de los videojuegos como medio de entretenimiento (Malone, 1981), y se ha visto especialmente por autores de referencia que han propugnado el uso de videojuegos como recursos de aprendizaje y que han sido inspiración para este trabajo: Marc Prensky (2004), Kurt Squire (2005) o James Paul Gee (2007).

Por otro lado, el creciente interés por el uso de los videojuegos en educación también ha dado lugar a la idea de usar elementos y mecánicas habituales de los videojuegos (tales como competiciones, niveles de experiencia, logros, rankings, etc.) dentro de otras actividades profesionales o educativas. A este tipo de uso de mecánicas de juego fuera de los propios videojuegos se le denomina gamificación (Deterding, Dixon, Khaled y Nacke, 2011). En el contexto educativo, esta idea se suele encarnas en modelos de aprendizaje en que los estudiantes se ven recompensados mediante “premios simbólicos” por sus logros conseguidos, habitualmente en forma de badges (insignias).

Han sido muchos los sitios de los que hemos tomado ejemplo, empezando por nuestro propio Campus virtual de Moodle, que introduce la posibilidad de gestión de insignias de usuarios así como la gestión de insignias de grupos (Moodle Pty Ltd, 2014). Las insignias de Moodle son compatibles con el sistema Mozilla Open Badges (Mozilla Foundation, 2014), en la que un usuario es reconocido ante un grupo de personas por todo aquello que ha ido aprendiendo ya sea de forma online u offline. El usuario irá coleccionando insignias en función de los logros obtenidos, de todo lo que haya prendido y de sus habilidades.

En el contexto particular de los videojuegos de entretenimiento, también existen sistemas que aplican técnicas de gamificación alrededor de los propios juegos. Este es el caso de los logros (achievements) de Xbox, que permiten calcular un GamerScore público asociado al perfil de cada jugador; de los trofeos (trophies) de Playstation; o igualmente las insignias (badges) que permiten aumentar el nivel de tu perfil en Steam. Todos estos mecanismos fomentan la interacción social más allá de los juegos individuales, dando lugar a plataformas de distribución de juegos gamificadas.

Inspirados en estos dos conceptos (uso de juegos en educación y creación de plataformas de juegos gamificadas), en este trabajo perseguimos diseñar una plataforma gamificada de juegos educativos, centrada en los contenidos de las titulaciones universitarias de Ingeniería Informática, y que haga uso de sistemas de puntos de experiencia, grupos de jugadores y obtención de insignias para fomentar nuevos modelos de educación competitivos.

En otras palabras, aspiramos a crear el Steam de los videojuegos educativos en informática.

## Sección 1.1. Planteamiento del trabajo

De cara a conseguir el objetivo propuesto, proponemos el desarrollo de la plataforma **OlimpiadaFDI**, donde se podrán enseñar conocimientos propios de nuestra carrera, Ingeniería en Informática, mediante la utilización de una suite de juegos educativos implementados en distintos lenguajes de programación (HTML5, Android y Java) para así animar a todos los estudiantes a hacerse un pequeño hueco en nuestra comunidad.

La idea detrás de este modelo sería que el conjunto de juegos forme un *ecosistema* en el que coexisten juegos de distintas plataformas, que enseñan distintas materias y que podrían ser desarrollados por distintas instituciones o grupos de trabajo. Nuestra visión incluye que, por ejemplo, distintas universidades pudiesen desarrollar juegos para aquellas áreas de la Ingeniería Informática en las que más destacan.

Para crear esta plataforma, el trabajo se divide en tres pilares fundamentales:

1) Creación de una infraestructura cuya base es un **servidor de backend** que ofrece una API de servicios REST para gestionar y almacenar datos de jugadores, juegos, insignias, puntuaciones y demás.

2) Diseño y desarrollo de un **portal web** en el que se podrá visualizar los resultados fruto de la interacción de los jugadores con los juegos, incluyendo sus perfiles, insignias obtenidas, gestión de grupos, rankings de jugadores, etc.

3) Creación de una **suite de juegos** de ejemplo, desarrollados con distintas tecnologías, pero todos ellos compatibles con la API de usuarios e insignias y que ejemplifican cómo futuros desarrolladores podrán valerse para integrar sus desarrollos en nuestro sistema.

Cabe señalar que hasta el comienzo del presente curso, desconocíamos algunos de los conceptos mencionados y otros que nos hemos ido encontrando a lo largo de nuestro camino; así como algunas de las tecnologías que hemos usado y finalmente utilizado. Por este motivo, el trabajo también incluye un estudio a fondo de las tecnologías empleadas, con especial atención a los requisitos para desarrollar juegos en plataformas móviles o en HTML5/JavaScript, así como a las plataformas gamificadas de juegos más usadas hoy en día.

## Sección 1.2. Estructura del trabajo

En el siguiente apartado, pasaremos a enumerar todos los capítulos de los que se compone esta memoria y daremos una breve explicación sobre cuál es la función y el propósito para cada uno de ellos.

Tras esta pequeña introducción inicial en la que analizamos cuáles fueron los motivos que nos llevaron a embarcarnos en este proyecto y, a grandes rasgos, su alcance y características principales, comienza el *Bloque I: Estudios previos y estado del arte*. Este bloque abarca el Capítulo 3, el Capítulo 4 y el Capítulo 5.

En el Capítulo 3 analizamos el impacto de los dispositivos móviles hoy en día y su importancia para el desarrollo de videojuegos, estudiamos las alternativas más populares a nivel de plataforma y comparamos las herramientas más comunes para desarrollar en ellas.

En el Capítulo 4 analizamos la evolución del desarrollo web y en concreto la tecnología HTML en su última versión para aplicar todo esto en el desarrollo de videojuegos en formato web.

En el Capítulo 5 realizamos un amplio repaso de las plataformas mencionadas en la *Introducción*, que tuvimos en cuenta y en las que nos fijamos a la hora de establecer la idea de nuestro proyecto así como la existencia de *badges*.

El *Bloque II: Descripción del sistema* está compuesto por el Capítulo 6 y el Capítulo 8.

En el Capítulo 6, y sin entrar todavía en términos muy técnicos, describimos los pilares más importantes que conforman nuestra plataforma: el servidor y los videojuegos. Hablamos de nuestra infraestructura como factor principal.

En el Capítulo 8 se realiza la especificación de todos los juegos que forman la suite y la web que hace de frontend intermediaria entre ambas.

El *Bloque III: Detalles de implementación* está compuesto por el Capítulo 8, el Capítulo 9 y el Capítulo 10.

El Capítulo 8 es donde el servidor toma el protagonismo, y se explica cómo ha sido diseñado, cómo se encarga de gestionar todos sus servicios y cómo hemos decidido integrar todos los componentes del proyecto en nuestra infraestructura.

A continuación, el Capítulo 9 y el Capítulo 10 describen a bajo nivel y en profundidad cómo se ha desarrollado tanto la web de OlimpiadaFDI como todo el conjunto de juegos en las diferentes plataformas de programación. Una valoración más técnica y precisa de su implementación, sus funcionalidades y cómo hemos logrado sacarlos adelante a pesar de las dificultades y hándicaps tecnológicos que encontramos.

El *Bloque IV: Caso de estudio y conclusiones* está compuesto por el Capítulo 11 y el Capítulo 12.

El Capítulo 11 se centra en simular todos los componentes de lo que sería el proceso completo que transcurre desde que un jugador se dispone a jugar hasta que sus insignias han sido registradas en la base de datos, guardadas por tanto en el servidor y finalmente reflejadas en la web. Este caso de estudio cuenta además con la participación de usuarios externos al proyecto que nos han dado su opinión y han prestado su ayuda para identificar nuevas oportunidades.

Destinamos el Capítulo 12 a realizar la reflexión del proyecto: Acerca de su elaboración y planteamiento inicial, la revisión de objetivos cumplidos, y conclusiones finales.

Reservaremos el Capítulo 14 para exponer la contribución individual de todos los miembros del grupo.

Por último, los capítulos finales de la memoria irán destinados a la *Bibliografía y referencias* así como al *Anexo*, que complementa el contenido de este documento.

# Chapter 2. Introduction

In the heyday of the virtual learning, student motivation is an essential component for the good performance. Didactic games appear as further improvement in this aspect, and bring about a change in the conventional paradigms of teaching.

Currently, the classical method of formal education seems to be overshadowed by the possibilities that new technologies bring. The formal method is oriented to individual study, and the trend is that everything becomes more and more social.

Both our curiosity about the world of teaching and education, together with the appeal that involves the use of educational games for this purpose, which we would serve as a channel of communication between the content and the student, aroused our interest and made us to decide on bringing together the knowledge acquired throughout our career, and the mechanics of learning.

Large precursors, masters of this art and people of great global influence in the field of learning based on video games, whose ideas we try to rely when making our work, are Marc Prensky, James Paul Gee and Kurt Squire, among others.

The essence of their ideology is based on a gradual educational reform consistent to current times. They offer a different perspective on the learning process, defending that nowadays teaching must be different from the past, motivating young people and increasing their passion for technology. Through video games, internet or mobile devices, ultimately, using technology as a tool for development. This type of education is not only good for content's teaching but also skills learning. In this way you can connect with students, make them to become interested and take the initiative to study and learn to be competent in the future.

> We need to discover the passion of each student.
>
> Teachers today should eliminate the masterclasses.
>
> *Marc Prensky* (June y November 2014)

Finally, to emphasize the idea of reward; the students win "Awards" for their achievements in playing, which we refer to as **badges** ("Insignias"). They might be compared with others from known systems for other platforms such as the **achievements** in **Xbox**, the **trophies** in **Playstation**, or equally **badges** that allow to increase the level of your profile on **Steam**.

This work, in short, has been the result of both, our growing interest in the use of games in education, to teach new students dynamic educational and didactic in which games are the main pillar of the self-study, and the need to create a publishing platform capable to manage them and make them available to all of us.

## Section 2.1. Work approach

The purpose of the project **OlympiadFDI** is teaching part of the knowledge of our career, Computing Engineering, using a suite of educational games implemented in different programming languages (HTML5, Android and Java) to thus encourage all students to be part of our community.

In order to this project to materialise it was necessary to create an infrastructure whose base is a server that offers REST services and in which we will keep and we will manage both data bases of players, their badges obtained in the games, and other critical information.

On the other hand, our project is based on a portal website where you can view the results as a result of the interaction of the players with games: all activity and score rankings of both, potential students as well as groups which can be formed.

Additionally, we can highlight the use of an own API that abstracts the functionality of our platform, in which future potential developers can use to integrate their developments in our system. These services will be described in detail later and during the Annex.

All this initial approach showed us to the concept of gamification of processes, which we wanted to collect and adopt as main idea and therefore translate it into our work, especially the gamification applied to education. We have taken examples from different sites, but firstly from our virtual Campus of Moodle, which introduces the possibility of users' and groups' badges management (Moodle).

Another platform following this initiative is Mozilla Open Badges (Open Badges), in which a user is recognised by a group of people for everything that has been learning either online or offline. The user will be collecting badges based on achievements, what has learnt and abilities. All these initiatives reflect the generational change that is taking place nowadays; a few years ago no one had proposed using games or videogames for educational and academic purposes.

Until the beginning of this year, we were unaware of some the concepts mentioned or others that we have come across along our way; as well as some of the technologies that we have found and finally used.

We could define our project as the **"Steam for educational games".**

## Section 2.2. Structure of the work

In the next section, we will enumerate all the chapters included in this memory and we give you a brief explanation about what is the operation and purpose for each one of them.

After this small initial introduction in which we analysed what were the reasons that led us to embark on this project and, broadly speaking, its scope and main features, we continue with Capítulo 3. In this chapter we look at the impact of mobile devices today, study the most popular alternatives at platform level and compare the most common tools for developing them.

In Capítulo 4, we analyse the evolution of web development and in particular the HTML technology in its latest version.

Capítulo 5 makes a broad review of the platforms mentioned above in the introduction, those which we had in mind, and in which we focussed to set up the idea of our project as well as the concept of **badges**.

In Capítulo 6, and without stepping into very technical terms, you will find material amount of information referred to one of the most important pillars of our platform: server and video games. We explain about our infrastructure as a major factor, a detailed specification of each of the games that form part of the suite and the web that makes frontend intermediary between the two.

Capítulo 8 is where the server takes the stage, and it explains how it has been designed, how manages all the services and how we have decided to integrate all the components of the project in our infrastructure.

Then, Capítulo 9 and Capítulo 10 describe at low-level and in depth how both, OlimpiadaFDI website and games have been developed in different programming platforms. A more technical and precise valuation of the implementation, its features and how have managed to bring them forward despite difficulties and technological challenges we encounter.

Capítulo 11 focuses on simulating all components of the full process that takes place since a player prepares to play until their badges have been registered in the B.D, therefore saved on the server, and finally shown on the web. This case study also has the participation of external users to the project that gave us their opinion and helped to identify new opportunities.

We dedicate Capítulo 12 to make the reflection of the project: about its preparation and initial approach, review of objectives, and final conclusions.

Capítulo 14 is dedicated to expose the individual contribution of all the members of the group.

Finally, the final chapters of the memory will go to the Bibliography and References as well as to Annex, supplementing the contents of this document.

# Bloque I
# Estudios previos y estado del arte

# Capítulo 3. Dispositivos Móviles

En este capítulo vamos a revisar las tecnologías, conceptos y herramientas disponibles para el desarrollo de aplicaciones móviles, prestando especial atención al desarrollo de juegos en general y juegos educativos en particular.

Para ello, empezaremos con una revisión de las distintas plataformas móviles (Sección 3.1), entre las que destacamos Android por ser la más abarcable para un desarrollo de código libre (Sección 3.2). Posteriormente, nos centramos en las distintas herramientas de desarrollo de juegos disponibles la plataforma Android (Sección 3.3).

Finalmente, en la Sección 3.4 presentamos las principales conclusiones derivadas de este estudio.

En nuestro, caso hemos justificado el desarrollo en dispositivos móviles haciendo referencia a conceptos innovadores, como **Mobile Learning**, una idea ya incluida en la idea más genérica **Ubiquitous Learning**. Estos términos hacen referencia a una nueva metodología de aprendizaje basada en el dinamismo y accesibilidad que proporcionan las nuevas tecnologías. Los alumnos comienzan a tener hoy en día la portabilidad y libertad para aprender sobre potencialmente cualquier tema, en cualquier momento y lugar, por ejemplo buscando información en Internet a través de un buscador, accediendo al campus virtual de su Universidad, o utilizando una de las aulas virtuales que proporcionan algunas webs de enseñanza online.

La idea detrás de **Ubiquitous Learning** en un futuro cercano es evolucionar estas ideas hasta el extremo, mejorar las plataformas de aprendizaje, e integrarlas en el entorno habitual del usuario hasta conseguir que estas sean invisibles al alumno; y haga uso de ellas sin siquiera ser consciente. Cuando esto se logra, los estudiantes aprenden siguiendo el hilo de sus propios razonamientos, casi satisfaciendo su propia curiosidad; acceden de forma inalámbrica, comparten sus conocimientos en su círculo de amistades y utilizan recursos en la nube.

## Sección 3.1. Plataformas móviles

Intuitivamente, todos conocemos el gran impacto que han generado hoy en día la proliferación de dispositivos móviles en nuestras vidas. Términos como conexión y accesibilidad nunca han estado tan en auge, y es que la penetración de los dispositivos móviles, en cualquiera de sus variantes, smartphones o tablets, ha sido indudable.

Las plataformas móviles ofrecen interesantes ventajas, tanto para los jugadores como para los desarrolladores.

A los usuarios les ofrece la inmediatez y comodidad de jugar con sus aplicaciones favoritas, o, en nuestro caso, juegos didácticos, cuando quiera. Esta conveniencia permite explotar al máximo los conceptos que estudiaremos con más profundidad más adelante, como **Mobile Learning**, un concepto que deriva de otro, **Ubiquitous Learning**, o el paradigma sobre el aprendizaje que sostiene que cualquier usuario puede aprender sobre cualquier cosa en todo momento y en cualquier lugar, debido, principalmente, a que las herramientas de enseñanza están tan integradas en la vida cotidiana, que el alumno no tiene siquiera la percepción de estar *aprendiendo*. Para que exista **Ubiqutous Learning**, las herramientas de aprendizaje deben ser accesibles e interactivas, así como adaptables e inmediatas (Core Education, 2013 y Education, 2025).

En el caso de los desarrolladores, los dispositivos móviles ofrecen una oportunidad de dar a conocer sus creaciones a una comunidad de jugadores en constante crecimiento, y de un perfil más amplio que los jugadores de consolas, como el mercado femenino, o el mercado más *maduro*, con una media de 43 años en el caso de juegos sociales, y con jugadores de incluso más de 60. Precisamente, estos perfiles conforman el

perfil **casual**.

Los *casual gamers* pueden referirse a *gamers* que juegan a *juegos casual*, o simplemente *gamers* que, en proporción, juegan menos que el resto. Curiosamente, los videojuegos móviles más rentables son **casual**, y son los pioneros de un sistema de financiación basado en **micro-compras** dentro de la propia aplicación. Este tipo de juegos móviles son los primeros en la listas de ingresos en Google Play y App Store.

Cabe destacar además que en 2013 los usuarios invirtieron más dinero en juegos móviles que videoconsolas (Puromarketing, 2013).

Hay una gran variedad de plataformas móviles disponibles para usuarios de todas las clases. A continuación enumeraremos los principales sistemas móviles.

### Sección 3.1.1. iOS

iOS es un sistema operativo basado en una variante del Mach kernel de Mac OS X, el cual, a su vez tiene una base UNIX. Es desarrollado por Apple Inc. para sus dispositivos móviles: iPod touch, iPhone e iPad.

iOS mantiene una arquitectura robusta y diferencia 4 capas de abstracción:

1. La capa del núcleo del sistema operativo,
2. La capa de "Servicios Principales",
3. La capa de "Medios de comunicación", y
4. La capa de "Cocoa Touch".

Todo el sistema se encuentra en el directorio "/root" del dispositivo y ocupa menos de 500 MB.

Apple ganó mucha popular en el periodo comprendido entre 2007 y 2010 debido principalmente a una serie de acertadas decisiones empresariales tomadas por el entonces CEO de la compañía, Steve Jobs (Figura 1), quien marcó una serie de pautas respecto al diseño e interacción humano-máquina que debían seguir los dispositivos móviles, y sembró las bases del diseño minimalista que siguen hoy en día casi todas las compañías.

**Figura 1: Steve Jobs, en la presentación del Iphone**

Es importante mencionar que el entorno de Apple es muy atractivo para los desarrolladores de videojuegos, ya que, según la web Gustavo Franceschini (2011), los usuarios de iOS son los que más tiempo pasan jugando, en comparación con Android, y además, son más proclives a invertir dinero en sus aplicaciones (Ameu8, 2014).

Apple ofrece una plataforma conveniente para los usuarios, donde las compras de aplicaciones o contenido multimedia (iTunes) están mucho más generalizadas que el resto de sistemas.

A los desarrolladores les ofrece herramientas de desarrollo propias, como XCode, la cual sólo está disponible para la plataforma MAC. Además, ofrece una mayor simpleza que el resto sistemas operativos móviles, principalmente debido a que existe un número limitado de dispositivos iOS (y resoluciones), y el fuerte versionado del sistema operativo en cada dispositivo.

Esto hace que el desarrollo sea más sencillo a la hora de crear interfaces de usuario (GUIs) que luzcan bien en cualquier dispositivo, o asegurarse un comportamiento consistente en todas las máquinas.

A pesar de ello, desechamos el desarrollo para el sistema operativo de Apple debido a que pocos de nuestros miembros tienen experiencia en el desarrollo para esta plataforma; y por, resultar algo restrictiva su programación nativa usando herramientas que imponen de forma inevitable una computadora o máquina virtual MAC OS (XCode).

Del mismo modo, el desarrollo en iOS es algo más tedioso debido al hecho de que sólo se pueden instalar tus nuevas aplicaciones en una serie de máquinas que debes *registrar* previamente.

Por último, cabe mencionar que los costes de publicación de software en el App Store pueden ser, en función de la licencia requerida, US$0, US$99/año (standard), ó US$299/año (enterprise).

### Sección 3.1.2. Android

Android es el sistema operativo de Google para móviles. Está basado en UNIX, y permite desarrollar con herramientas libres, como Eclipse, IntelliJ, o, actualmente, Android Studio, la herramienta de desarrollo *oficial* (Intel, 2015).

Como es el sistema operativo que hemos escogido, nos centraremos, en este punto, en las diferencias con respecto al resto de sistemas. Posteriormente, en el capítulo 2.2, realizaremos un estudio más profundo de la tecnología.

Las principales diferencias para los desarrolladores de esta tecnología con respecto a iOS, su competidor más directo, son el lenguaje de programación nativo, la plataforma de desarrollo, el sistema de permisos de las características internas del móvil, la variedad de dispositivos y resoluciones para las que programar las aplicaciones, o las distintas versiones que contemplar del sistema operativo (APIs). Visualizar la Tabla 1

Incluso entonces, más allá del mero hecho de desarrollar, existen otras diferencias significativas a tener en cuenta; algunas de las cuales ya hemos mencionado, como el perfil de los usuarios ambas plataformas, o el perfil de usuario del potencial jugador (el *target*), el sistema de monetización (aplicación gratuita/de pago, sistema de micro transacciones internas, y combinaciones entre ellas), o simplemente, la tecnología a utilizar.

Sin embargo, no hemos mencionado un aspecto importante que debería al menos contemplarse aunque sólo sea de manera superficial; y es la plataforma de publicación (Tabla 2).

Sólo habiendo realizado este estudio previo, podemos comenzar nuestro desarrollo.

**Tabla 1: Principales diferencias, en cuanto a tecnología, entre Android e iOS. (Schroeder, Thüs y Amine Chatti, 2014)**

| | Android | iOS |
|---|---|---|
| Language | C, C++, Java | C, C++, Objective-C |
| Manufacturer | Various | Apple |
| Resolution | 240x320 ... 1440x2560 (var.) | 320x480 ... 1536x2048 (5) |
| Versions | 8 in use (2.2 to 4.4.2) | 3.1.3, 4.2.1, 5.1.1, 6.1.6, 7.1 |
| Source model | Open Source | Proprietary EULA |
| OS family | Linux | Mac OS X |
| Storage | Internal, SD card | Internal, wireless |
| Permissions | 145 different | Less, on demand |
| Free Development | Android Emulator, Phone | iPhone simulator |

**Tabla 2: App Store vs. Google Play (Schroeder, Thüs y Amine Chatti, 2014).**

| | iOS | Google Play |
|---|---|---|
| Introduced | July 2008 | March 2009 |
| Apps | 1,157,279 (04.04.2014) | 1,192,749 (03.04.2014) |
| Downloads (May 2013) | 50,000,000,000 | 48,000,000,000 |
| Revenue per day | $ 5.1 million | $ 1.1 million |

## Sección 3.1.3. Windows Phone

Windows Phone, o WP para abreviar, es el Sistema Operativo de Microsoft para dispositivos móviles. Está basado en Windows Mobile y Zune, y fue lanzado al mercado en Octubre 2010. La interfaz gráfica está basada en el diseño de Metro (Figura 2), del que Windows 8 hace uso.

A lo largo de sus actualizaciones, WP ha ido actualizándose para dar soporte a nuevas funcionalidades y mayor personalización del sistema por parte de sus usuarios. A pesar de sus esfuerzos, muchas de sus ideas ya existían en dispositivos Android antes de su aparición en Windows Phone, como el sistema de notificaciones, o personalización de la imagen de fondo.

Uno de los pilares más atractivos de esta plataforma, sin embargo, es el asistente por voz, Cortana (una referencia a la IA de la saga Halo, también de Microsoft). Cortana lleva a cabo búsquedas por voz, y permite un alto grado de interacción. Según Whitman, Microsoft ha realizado un gran trabajo en este aspecto, aportando enormes avances en los campos de Aprendizaje Automático y Reconocimiento de Voz (Ryan Whitwam, 2014).

Mientras que el asistente de voz que implementa Android es algo más impersonal, y realiza sus búsquedas a través de Google, Cortana pretende ser más *humana* (cuenta incluso chistes) y conversacional, y utiliza el motor de búsqueda de Microsoft, Bing.


**Figura 2: Imágenes del interfaz gráfico de Windows Phone 8.1 basado en Metro.**

## Sección 3.1.4. Blackberry OS

Blackberry OS es el sistema operativo desarrollado por Blackberry para dispositivos handheld, (literalmente, dispositivos que caben en una mano, como los teléfonos móviles smartphones). La característica principal de

este sistema es que permite la sincronización del software corporativo, como Microsoft Exchange o Lotus Notes mediante MIDP, Mobile Information Device Profile, una especificación para desarrollos Java en plataformas móviles.

MIDP forma parte de Java ME (Micro Edition), que está integrado en el hardware de algunos dispositivos móviles, y fue lanzado por vez primera en Abril del 2001.

### Sección 3.1.5. Otros

Existen otros Sistemas Operativos móviles en el mercado, menos prominentes y con muchos menos usuarios que los ya mencionados. Estas son algunas plataformas dignas de mención:

**Symbian** (Figura 3), el sistema operativo de Nokia, ya desaparecido. Los orígenes de Symbian se remontan a mediados de los 90, época de PDAs y Handhelds. Está basado en el sistema EPOC, el cual fue desarrollado por **Psion**, para estos dispositivos.

Fue una plataforma muy prometedora hasta 2007. La llegada del iPhone supuso una auténtica revolución, un cambio inesperado, y no todas compañías supieron adaptarse a los nuevos tiempos. Tras la presentación del iPhone, la respuesta de Symbian fue escasa y tardía, y no sólo eso, sino que, como en el caso, tampoco consiguió seguir la tendencia que marcó Apple (Hipertextual, 2014).

Además, la colaboración de Nokia con Microsoft, delegó a Symbian a un segundo plano, por detrás de Windows Phone.


**Figura 3: Imágenes de dispositivos de Nokia con Symbian**

**WebOS**, una plataforma desarrollada por **Palm Pre**. El diseño del interfaz de usuario, basado en *tarjetas* (Figura 4), se llevó a cabo por Matías Duarte, actualmente director de experiencia de usuario de Android. Palm Pre fue adquirido por la compañía **HP**, la cual decidió usar el sistema WebOS como sistema operativo para impresoras y PC's.

Durante un tiempo no hubo cambios relevantes, y no fue hasta el 2009, año en el que Léo Apotheker, por entonces CEO de HP, fue sustituido por Meg Whitman, ex-CEO de eBay, cuando se liberó el código de esta

plataforma, creando así Open WebOS.

Finalmente, LG compró el proyecto WebOS a HP y lo utilizó como sistema operativo para *Smart TVs*, manteniendo, eso sí, el aspecto por *tarjetas* que trajo consigo el WebOS original (Hipertextual, 2014).


**Figura 4: Imágenes del Touchpad HP, con el sistema WebOS**

## Sección 3.2. A fondo: la plataforma Android

Como ya comentamos, centramos nuestros esfuerzos sobre la plataforma Android, utilizando únicamente tecnología nativa.

Android es un sistema operativo inicialmente pensado para teléfonos móviles, al igual que iOS, Symbian y Blackberry OS. Lo que lo hace diferente es que está basado en Linux, **un núcleo de sistema operativo libre**, gratuito y multiplataforma. La empresa Google ha sido el mayor contribuyente y promotor de este software, y hoy en día es el sistema operativo móvil más popular en todo el mundo, lo que deriva en el estereotipado de sus usuarios (Figura 5).

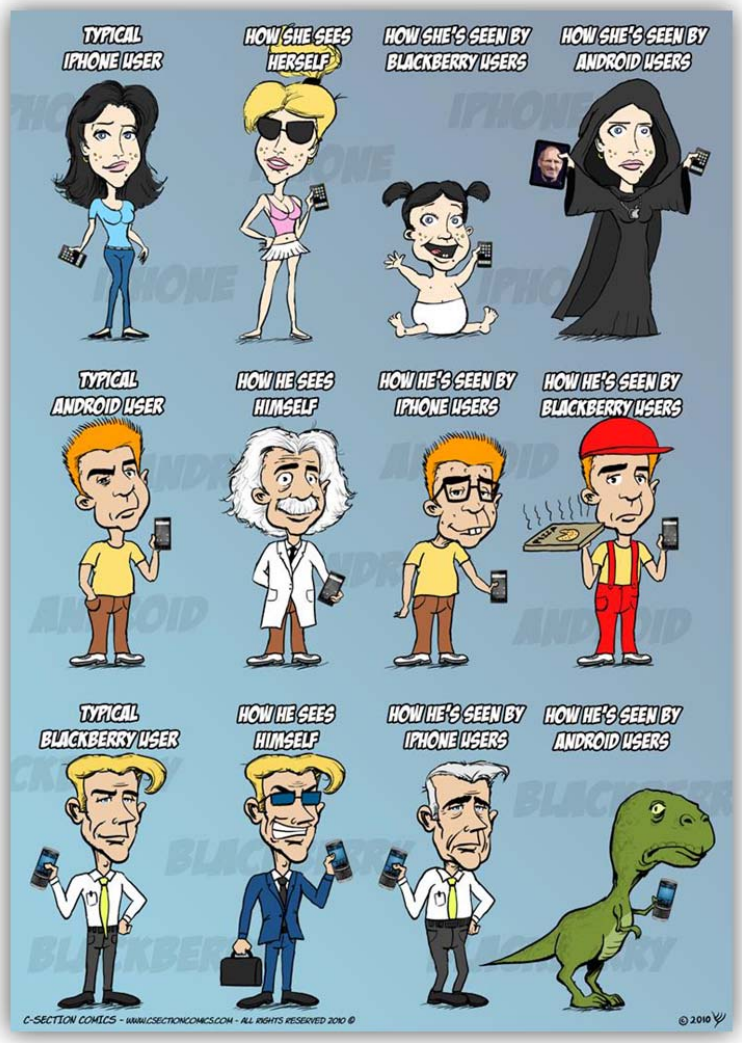
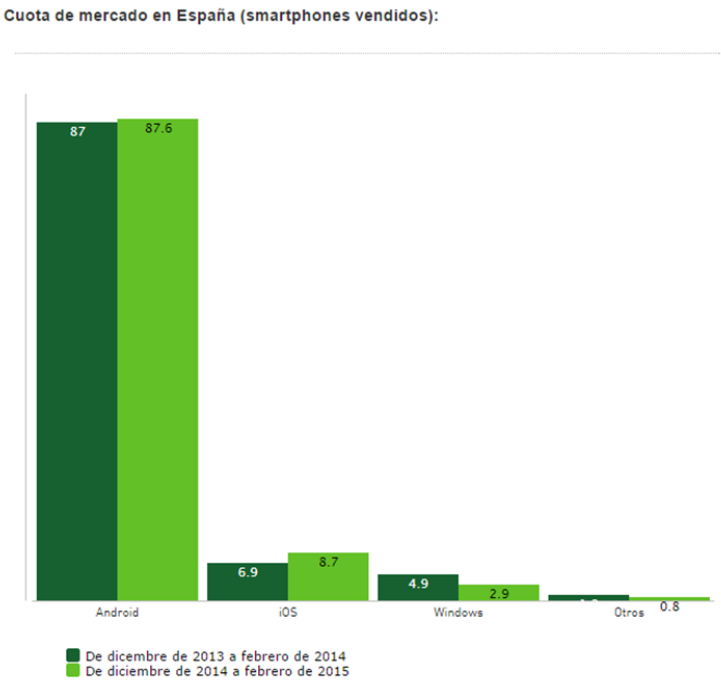



**Figura 5: Cómo se ven los usuarios de las distintas plataformas móviles**

Una de las mejores características de este sistema operativo es que es completamente libre. Es decir, ni para programar en este sistema ni para incluirlo en un teléfono es necesario pagar nada. Y esto lo hace muy popular entre fabricantes y desarrolladores, ya que los costes para lanzar un teléfono o una aplicación son muy bajos.

Nos hemos decantado por esta tecnología debido a su proyección de futuro y su enorme presencia en el mercado actual en lo que a dispositivos móviles se refiere: Incluso a pesar del enorme crecimiento de Windows Phone, el cual se prevé superar a iOS este año 2015 (Tabla 3), Android alcanza una cifra superior al 58% de porcentaje de mercado global. Otras virtudes del SO de Google son su versatilidad, su facilidad de uso, y su detallada documentación.

**Tabla 3: Porcentaje de mercado de los últimos años en Sistemas Operativos móviles en España (2015)**



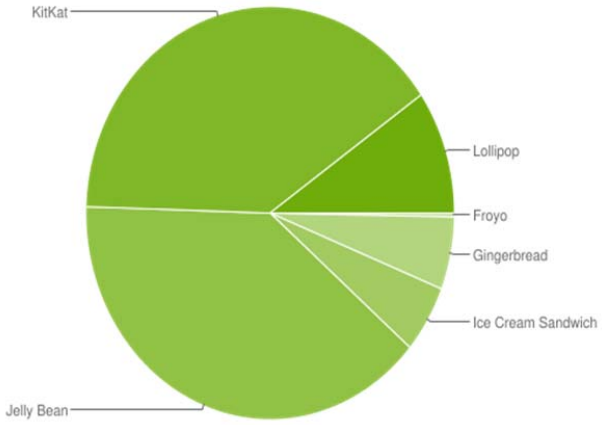

No sólo es un sistema operativo robusto y eficaz, sino que además trae consigo una gran cantidad de herramientas de ayuda a los desarrolladores, como un framework de desarrollo propio, el Android Studio, el cual se encuentra en su versión 1.2 a fecha de Junio 2015.

Digno de mención es la pequeña revolución interna que hubo al comienzo de la realización de este proyecto, entre Septiembre 2014 - Enero 2015. Hasta antes de Septiembre, el desarrollo sobre Android se realizaba de distintas formas, entre ellas, la más popular era una serie de plugins o extensiones para la herramienta de desarrollo en Java Eclipse. Fue en torno a esa fecha cuando Google decidió dejar de dar soporte a esas extensiones para fomentar la migración de los desarrollos Android a Android Studio, la nueva herramienta de desarrollo específica para Android impuesta por Google.

Una vez seleccionado el Sistema Operativo objetivo, Android, hay aún una cuestión importante a la hora de desarrollar para esta plataforma, y es el nivel de la API o SDK destino de la aplicación. En nuestro caso, todos nuestros juegos tienen fijada un nivel API mínimo de 15, equivalente a una versión android 4.0.3, también conocida como Ice Cream Sandwich; y un nivel de API deseado 21, o 5.0 Lollipop (Tabla 4, documentación Android Studio).

**Tabla 4: Listado y gráfica de las distintas versiones de Android existentes (Android Developers, 2015)**

| Version | Codename | API | Distribution |
|---|---|---|---|
| 2.2 | Froyo | 8 | 0.3% |
| 2.3.3 - 2.3.7 | Gingerbread | 10 | 5.7% |
| 4.0.3 - 4.0.4 | Ice Cream Sandwich | 15 | 5.3% |
| 4.1.x | Jelly Bean | 16 | 15.6% |
| 4.2.x | | 17 | 18.1% |
| 4.3 | | 18 | 5.5% |
| 4.4 | KitKat | 19 | 39.8% |
| 5.0 | Lollipop | 21 | 9.0% |
| 5.1 | | 22 | 0.7% |




Para evitar errores, siempre es recomendable compilar las aplicaciones en la versión más actualizada posible. Nosotros compilamos todas nuestras aplicaciones con un nivel 21.

Con esta configuración, aseguramos que nuestras aplicaciones sean **plenamente funcionales con el 90,4% de los dispositivos Android** que existen en la actualidad; o lo que es lo mismo, **el 52,52% de la totalidad de dispositivos móviles del mundo**, a fecha de marzo 2015.

Android Studio es un entorno de desarrollo integrado para la plataforma Android basado en IntelliJ IDEA de JetBrain. Su primera versión beta 0.1 fue lanzado en Mayo de 2013, y ha recibido actualizaciones de una manera más o menos periódica desde entonces, hasta la versión actual, 1.2, a fecha de Mayo 2015.

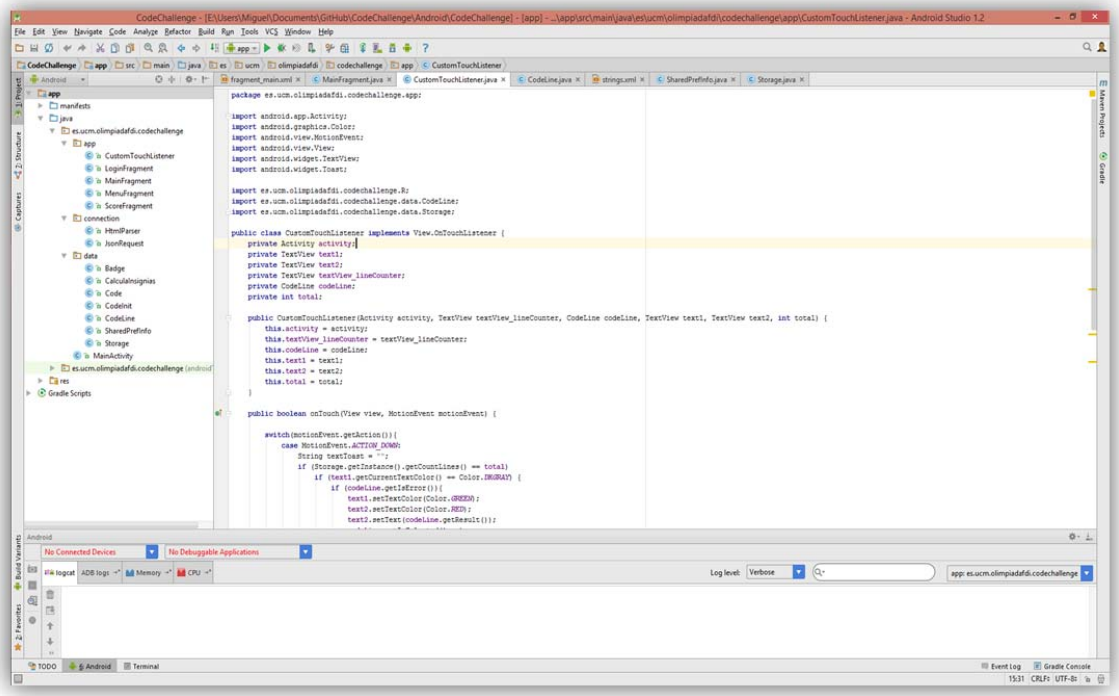


**Figura 6: Interfaz de usuario de la herramienta de desarrollo Android Studio v1.2.**

## Sección 3.3. Herramientas de desarrollo para Android

Para ayudarnos en el desarrollo Android, podemos valernos de alguna de las herramientas de desarrollo existentes. A continuación mostramos las más prometedoras:

Utilizamos la tecnología Adobe AIR, un lenguaje sencillo y orientado a objetos que se utiliza a través de la herramienta Adobe Flash CC; la cual está disponible para Windows. Uno de los factores más atractivos es la facilidad para crear interfaces adaptativas (responsive), y su facilidad de exportación a web, así como a aplicación pesada o móvil. Además, extendía sus posibilidades en cuanto a animación y diseño gráfico ya que se integraba con otras herramientas de Adobe, como Photoshop o Illustrator.

Motores de juego. Entre ellos GameMaker (GameMaker, 2015), Construct 2 (Construct 2, 2015), Unity (Unity, 2015), o Game Salad (Game Salad, 2015). Los motores de juego son herramientas específicas para la creación de videojuegos. Normalmente abstraen una serie de funcionalidades que permiten a los desarrolladores centrarse en la lógica de su juego, sin tener que enfrentarse al reto que supone la creación de físicas, o implementación de los eventos que tienen lugar. Facilitan el trabajo haciendo este tipo de cosas invisibles a los programadores, los cuales sólo tienen que parametrizar la lógica y centrarse en el diseño visual y jugable de sus desarrollos.

## Sección 3.4. Conclusiones

En este capítulo hemos visto las plataformas más comunes en relación a dispositivos móviles, y algunas herramientas que permiten a los desarrolladores crear juegos o aplicaciones interactivas para estas plataformas.

Nos hemos centrado en el desarrollo para Android, debido a que es la plataforma más conveniente dadas nuestras circunstancias.

También hemos visto motores de juego y su uso en el desarrollo de juegos móviles, y nos centramos en el GameMaker, aunque con resultados negativos, como veremos más adelante.

La evolución de las tecnologías es importante para llevar a cabo este paso, por eso hemos estudiado las plataformas más importantes y las herramientas más comunes para desarrollar en ellas. Como parte de este estudio, hemos observado también el rápido crecimiento que están teniendo los juegos construidos con HTML5 y JavaScript. En el siguiente capítulo, exploraremos en mayor profundidad las tecnologías y plataformas existentes para el desarrollo de juegos en HTML5.

# Capítulo 4. HTML5

En este capítulo se revisarán conceptualmente las características de la última norma vigente de HTML, utilizada tanto para la implementación de juegos como para el frontend de la plataforma, prestando especial atención al nuevo elemento para dibujo de gráficos, canvas, incluido en esta versión.

Para ello, se comenzará con una introducción consistente en la evolución de HTML desde sus comienzos hasta el día de hoy (37Sección 4.1), incluyendo las características que aportan JavaScript y CSS en su combinación con el mismo. Seguidamente, se incluirá un histórico de cambios la nueva versión con el último estándar obsoleto a día de hoy, HTML4 (Sección 4.2), seguido de una sección que explicará el nuevo elemento, canvas, incluido en la nueva versión (Sección 4.3). También estudiaremos el framework Bootstrap, utilizada para ayudar a definir el estilo en la página web. Finalmente en la Sección 4.5 se presentarán las principales ventajas que ofrece HTML5 para el desarrollo de videojuegos y se terminará el capítulo con un apartado de conclusiones.

## Sección 4.1. Introducción

HTML (Lenguaje de marcación de Hipertexto, *Hyper Text Markup Language* en inglés) es el lenguaje de marcas de texto de mayor utilización en Internet. No es propiamente un lenguaje de programación, sino un sistema de etiquetas. Así, no presenta ningún compilador, por lo tanto frente a errores de sintaxis trata de mostrar resultado en la visualización en lugar de marcar el error correspondiente.

Es creado en 1986 por el físico Berners-Lee, basado tanto en el concepto de hipertexto o link, el cual permite conectar dos elementos entre sí y el Lenguaje Estándar de Marcación General (SGML) basado en etiquetas que permite colocar etiquetas sobre un texto para indicar cómo será su visualización. Tras finalizar el desarrollo de su sistema y después de unir sus fuerzas con el ingeniero de sistemas Robert Cailliau, lo presenta a una convocatoria organizada para desarrollar un sistema de hipertexto para Internet, resultando finalmente la propuesta ganadora, llamada WorldWideWeb (W3) (LibrosWeb, 2015).

A lo largo de los años se estandariza el lenguaje HTML respecto a unas normas definidas por el WWW Consortium (W3C) desde 1996, después de que el primer estándar oficial (HTML 2.0, debido a que ni HTML ni su versión HTML+ son debidamente estandarizadas) se lleve a cabo por parte del organismo IETF (Internet Engineering Task Force)

La versión definitiva del estándar actual, HTML5, se publica en octubre de 2014.

Las limitaciones iniciales de HTML propician que varias compañías tengan que desarrollar nuevos lenguajes y programas para agregar nuevas características a la web no son implementadas con anterioridad. Así, algunos de estos desarrollos se convierten en accesorios populares y poderosos, de manera que su uso se extiende entre los desarrolladores web. Java y Flash son dos claros ejemplos.

Con el paso de los años y la masificación de usuarios en la red, Java y Flash empiezan a mostrar ciertas carencias dada su concepción como complementos (plug-ins), que provoca la no existencia de comunicación o integración entre aplicaciones y documentos.

Existe JavaScript, lenguaje interpretado incluido en los navegadores, que mejora experiencia de los usuarios y otorga funcionalidad a las páginas. Sin embargo, dado que no puede reemplazar funcionalidad de Flash y/o Java, su popularidad no es alta, dado que Java y Flash siguen siendo los únicos capaces de otorgar algunas funciones esenciales en la web como la reproducción de video. Así, durante cierto periodo de tiempo, el uso de JavaScript no es frecuente.

La mejora de los navegadores con el paso del tiempo provoca una mejor interpretación de JavaScript. Tras un aumento de usuarios de la red, parece que ni Java ni tampoco Flash van a poder proveer herramientas que demandan las aplicaciones. Toma fuerza JavaScript, siendo de gran importancia en el desarrollo de la web. Junto a HTML y CSS se convierten rápidamente en la mejor combinación para la evolución de la web (ver Figura 7).

La llegada de HTML5 supone la estandarización real de cada aspecto de la web. Cada tecnología obtiene un propósito claro. Así, HTML provee elementos estructurales, CSS se encarga de hacer la estructura utilizable, así como atractiva a la vista y JavaScript da dinamismo y permite construcción de aplicaciones web totalmente funcionales. (Gauchat, 2012)

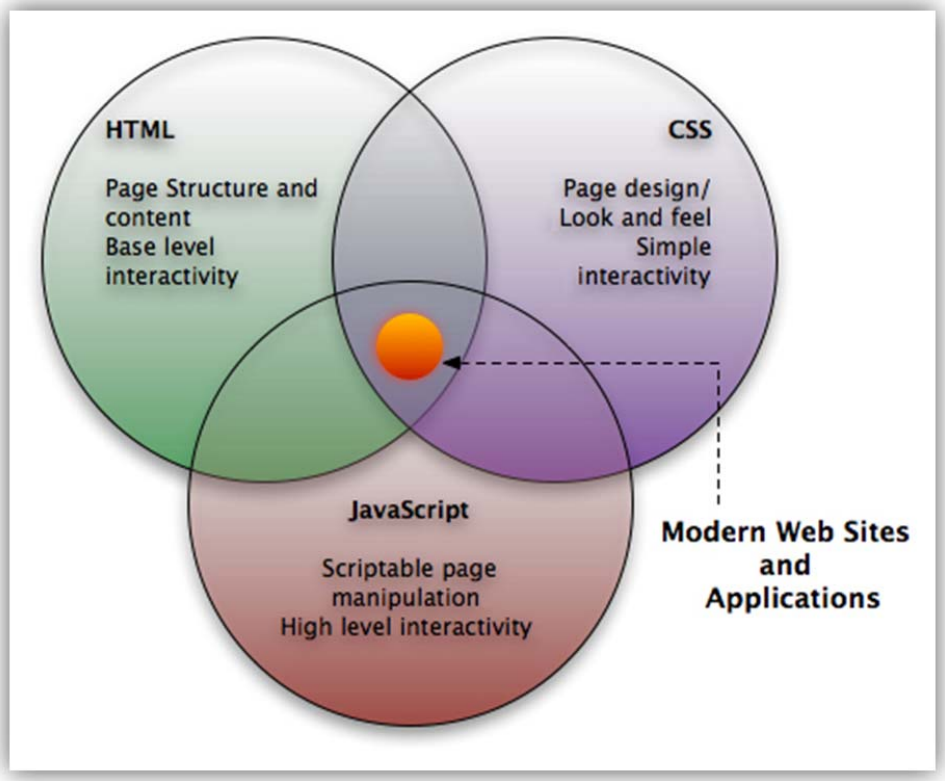



**Figura 7: Balance de tecnologías JavaScript, HTML y CSS en las web modernas. (99points, 2010)**

## Sección 4.2. Evolución hasta HTML5

### Lista de etiquetas agregadas y utilidad:

En HTML5 se agregan nuevas etiquetas con respecto a la versión anterior del estándar, HTML4. La principal novedad es la inserción del elemento canvas, al que dedicaremos una subsección en este mismo capítulo y permite creación y modificación de imágenes dinámicamente, aunque no es la única. En la tabla a continuación se listarán todas las nuevas etiquetas agregadas:

Tabla 5: Etiquetas nuevas HTML

| Etiqueta | Utilidad |
|---|---|
| <article> | Declara contenido como un artículo que tiene significado por sí mismo. |
| <aside> | Declara contenido que no tiene mucha relación en cuanto a significado con el resto de la página. |
| <audio> | Etiqueta estándar para insertar elementos de audio. |
| <canvas> | Utilizado para implementación y dibujo de gráficos e imágenes dentro de una página. |
| <datalist> | Define una lista de entradas (inputs) predefinidas. |
| <details> | Especifica detalles que el usuario puede escoger visualizar o esconder. |
| <dialog> | Utilizado para representar conversaciones. |
| <embed> | Define link a una aplicación externa o plug-in. |
| <figure> | Asocia con un caption un contenido incrustado. |
| <footer> | Contiene información sobre autor y sus derechos. |
| <header> | Contiene la cabecera del programa. |
| <mark> | Define texto remarcado. |
| <meter> | Representa medidas en rangos conocidos. |
| <nav> | Representa sección de la página orientada a navegación. |
| <output> | Indica el tipo de salida que se producirá con las acciones de la página. |
| <progress> | Indica el progreso que lleva completado determinada tarea. |
| <ruby> | Notación ruby. |
| <rp> | Decide qué mostrar cuando el navegador no acepta notación ruby. |

| | |
|---|---|
| <rt> | Explica el caracter o su pronunciación (referente a notación ruby). |
| <section> | Indica secciones genéricas. |
| <source> | Especifica varios recursos multimedia. |
| <time> | Muestra fechas o tiempos. |
| <video> | Etiqueta estándar para insertar elementos de audio. |

## Lista de etiquetas obsoletas de HTML4 y utilidad

Algunas de las etiquetas aceptables en HTML4 han pasado a estar obsoletas. El principal motivo es que muchas de ellas eran utilizadas para estilos concretos, pero el crecimiento en cuanto a utilización y el éxito que está teniendo CSS en la web ha provocado que muchos no sean realmente necesarios ya que CSS tendrá la misma funcionalidad que los mismos. La tabla siguiente refleja todas las etiquetas que quedaron obsoletas en la antigua versión del estándar:

**Tabla 6: Etiquetas obsoletas respecto a la versión anterior**

| **Etiqueta** | **Utilidad** |
|---|---|
| <acronym> | Define un acrónimo. |
| <applet> | Define un applet. Sustituida por embed. |
| <basefont> | Especifica ciertas funciones para el estilo de texto. Obsoleta frente al uso de CSS. |
| <big> | Texto de mayor tamaño. Preferible el uso de CSS. |
|  | Centra el texto. Preferible el uso de CSS. |
| <dir> | Lista títulos de un directorio. |
| <font> | Define la fuente del texto. Preferible el uso de CSS. |
| <frame> | Define una ventana particular dentro del frameset. |
| <frameset> | Define un conjunto de marcos. |

| | |
|---|---|
| <hgroup> | Se añadió a HTML5 pero pasó a estar obsoleta directamente debido al uso de header. |
| <noframes> | Utilizada para aquellos navegadores que no soportaban el uso de marcos. |
| <s> | Utilizada para texto irrelevante. |
| <strike> | Para texto tachado. Preferible CSS. |
| <tt> | Da formato al texto. Preferible uso de CSS. |
| <u> | Formato de texto subrayado. Preferible CSS. |
| <xmp> | Para formato de texto preformateado. Preferible CSS. |

## Lista de etiquetas modificadas

Algunas etiquetas han cambiado su utilización de una versión a otra, alguna mínimamente manteniendo su funcionalidad (input), otras por el éxito de CSS han visto limitada su funcionalidad. La tabla de todas las etiquetas modificadas se adjunta a continuación:

**Tabla 7: Etiquetas eliminadas**

| **Etiqueta** | **Utilidad** |
|---|---|
| <a> | Ahora solo se permite su uso como hyperlink. |
| <cite> | Se usa como título de un trabajo, respecto a su anterior uso como citación. |
| <hr> | Se le da una interpretación semántica al salto de línea. |
| <i> | Antaño daba formato de texto en cursiva, en la actualidad se puede dar formato a un párrafo i por CSS. Se utiliza como la etiqueta p, pero para párrafos concretos que necesitan diferentes estilos. |
| <input> | Se añaden nuevos atributos en HTML5 |
| <b> | Antaño daba formato de texto en negrita, en la actualidad se puede dar formato a un párrafo i por CSS. Su utilización actual es similar a la etiqueta i. |

## Sección 4.3. El elemento canvas

El elemento canvas (que se traduce al español como *lienzo*) se define como un entorno para la creación de imágenes dinámicas. Ofrece como principal ventaja la modificación de imágenes en tiempo real, además de que está integrado directamente sobre HTML de manera que no es necesaria la carga de un elemento o plugin de JavaScript. Además, JavaScript permite la modificación de los lienzos por lo que se mantiene la estructura HTML-CSS-JavaScript descrita anteriormente en la que cada uno de ellos mantiene su funcionalidad.

La accesibilidad en los navegadores, al estar integrado en HTML5, es superior al obsoleto y otrora muy utilizado flash.

Un Canvas es un lienzo de mapa de bits que puede representar imágenes o gráficos. Se dibuja sobre Canvas mediante la API Canvas 2D, que provee a Canvas de los métodos, objetos y propiedades para la creación de elementos y su visualización en dos dimensiones. Dentro de este lienzo tenemos posicionamiento de imágenes con coordenadas de manera que la esquina superior izquierda del lienzo representa la posición x=0, y=0 y ascienden los valores de x e y acorde a la siguiente imagen:

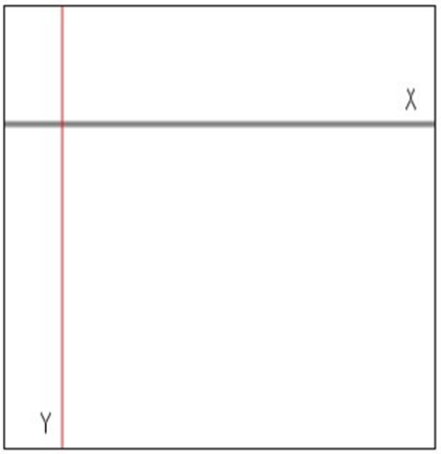



**Figura 8: Canvas 2D HTML5**

La ventaja que ofrece la API 2D de Canvas respecto al uso de la etiqueta img tradicional es que la imagen puede estar dentro de un lienzo y compartirlo con otras imágenes diferentes, lo que permite una simplificación a la hora de la realización de videojuegos. Es conveniente tener en cuenta algunas consideraciones, como serán:

- Para diferentes lienzos se declaran contextos (2D por lo general) por separado
- Insertando imágenes en un lienzo, se debe tener en cuenta el tamaño del lienzo.

**Ejemplos de uso de canvas:**

En los siguientes ejemplos, en los que incluiremos el código incluido para la formación del canvas a partir de dos imágenes, se observa la limitación a tener en cuenta del tamaño del lienzo.


**Figura 10: Ejemplo de canvas bien configurado**

**Figura 9: Código para la formación del canvas**

aparecen juntas y completas. Sin embargo, si la imagen es de un tamaño superior al canvas, para un código

```
<canvas id="myCanvas" width="600" height="400"></canvas>
<script>
var canvas = document.getElementById('myCanvas');
var context = canvas.getContext('2d');
var imageVader = new Image();
var imageBobby = new Image();
imageVader.onload = function() {
context.drawImage(imageVader, 0, 0);
};
imageBobby.onload = function() {
context.drawImage(imageBobby, 10, 10);
};
imageVader.src = 'Vader.jpg'
imageBobby.src ='BobbyTables.jpg'
</script>
```

similar pero un tamaño de canvas menor, podemos obtener un resultado no deseado:


**Figura 11: Ejemplo de canvas mal configurado**

En la imagen superior se observa un mal uso del Canvas, ya que el lienzo es inferior en tamaño a la imagen en segundo plano, por lo que el texto no se visibiliza de manera completa. Es una imagen de un Canvas de ejemplo en el que es observable que dos imágenes diferentes se pueden dibujar en el mismo lienzo.

Con respecto al desarrollo de juegos, la adición de canvas tiene un impacto directo, de manera que con ellos se pueden generar nuevos *sprites* (mapa de bits en movimiento), de manera que aportan una simulación de mayor realismo para los personajes o los escenarios y la interacción con el juego, aportando movimiento a los jugadores como a los sprites del juego.

## Sección 4.4. Bootstrap

Bootstrap es un framework de software libre creado por Twitter que se aprovecha de una combinación de herramientas HTML con plantillas CSS, utilizadas para un desarrollo web ágil.

En este proyecto se aprovechan algunas de sus ventajas:

*Sintaxis HTML:* Mantiene la estructura del flujo de programa, siendo muy sencillo de adaptar al código.

*Diseño en malla:* Permite crear una estructura de la web de manera que la distribución del contenido se vuelve flexible, a la par que adaptable a los navegadores.

*Diseño responsive:* adapta el diseño de la página web a cualquier dispositivo.

*CSS incorporado:* A pesar de que se usa CSS en bastantes partes del proyecto, la inclusión de uno propio por parte de Bootstrap permite adaptar la página web a los diseños más utilizados en la web sin necesidad de la creación de uno propio: utiliza un preprocesador llamado Less, que posee variables y funciones que aumentan la funcionalidad de CSS. Así, permite redimensionar los objetos y configurar el modo de su redimensión, lo cual es muy útil para tener diferente visualización de las páginas de una web sobre pantallas diferentes (Bootstrap settings, s.f.).

## Sección 4.5. Ventajas que nos ofrece HTML5

Se decide la utilización de HTML5 debido a la lista de ventajas que ofrece, tanto para la implementación de los juegos, como para el desarrollo de la página web. Entre ellos destacamos:

*Nativo:* HTML5 es nativo y, por lo tanto, independiente de terceros y plugins.

*CSS adaptable:* Permite compatibilidad con navegadores móviles, lo cual es de gran interés respecto al uso de la página web.

*Canvas:* Permite el uso y modificación de imágenes con mayor potencia que la etiqueta img, dando efectos visuales a las mismas.

*Proceso de Desarrollo Rápido:* No hay que esperar por una compilación, actualizaciones o depuración en tiempo real, por lo que una vez que el juego está finalizado se puede lanzar la ejecución directamente.

*Simplicidad de código:* permite hacer páginas más ligeras que se cargarán más rápidamente favoreciendo la usabilidad.

## Sección 4.6. Conclusiones

En este capítulo se ha realizado un repaso de las principales características de la herramienta HTML5, estándar actual de referencia para la elaboración de páginas web.

Durante el mismo, se han incluido las herramientas que fueron de mayor utilidad en el proyecto, así como se ha comentado en qué partes del proyecto ha sido utilizado, como son el portal de la Olimpiada FDI y los juegos. Además, se han expuesto las ventajas que nos ofrecía el desarrollar ambas con el uso de HTML5. Con ello se explicó a su vez el elemento canvas, la novedad de mayor importancia en lo referente a la manipulación de imágenes, vital para el desarrollo de videojuegos para plataforma web. Un apartado fue dedicado a Bootstrap, framework de software libre cuya funcionalidad permite la colocación eficiente de los bloques de una página web.

Con este capítulo termina el apartado de investigación acerca de las tecnologías. Daremos paso en el próximo a una explicación de las plataformas de servicios multijugador en las que nos hemos inspirado para la implementación de nuestra propia plataforma.

# Capítulo 5. Plataformas de servicios multijugador

En este capítulo se describen las plataformas de servicios multijugador, en las que se ha inspirado nuestro proyecto. Se realizará un estudio de las más famosas a nivel internacional.

Para ello, empezamos con una definición en la que explicamos qué son estas plataformas (Sección 5.1) seguido de una revisión de las distintas plataformas de renombre (Sección 5.2). Para terminar el capítulo, tendremos una sección en la que explicaremos la influencia de estos trabajos sobre el nuestro (Sección 5.3)

## Sección 5.1. Definición

Son plataformas cuyo objetivo es la interacción del jugador con diferentes juegos y los servicios que pueden ofrecer. Entre ellos, se encuentran: jugar a los juegos incluidos en la plataforma, descargar los mismos o actualizar los juegos en caso de ser desarrollador, entre otros.

Una de las grandes ventajas de las plataformas de servicios multijugador en cuanto a entretenimiento es la interacción entre los usuarios. Así, estas plataformas son conocidas no solo por ofrecer entretenimiento en cuanto a juegos sino por la definición de algunos objetivos en los mismos y la posibilidad de obtención de recompensas al lograr alcanzarlos. Estos objetivos, sumados a la implementación de elementos en la plataforma, como clasificaciones, grupos y chats, permiten una interacción más amena entre los usuarios del sistema.

En cuanto a las clasificaciones, cada una de las plataformas expuestas a continuación tiene su propia visión, pero todas coinciden en la implementación de una sección de logros por cada videojuego, opcional u obligatoria para los desarrolladores según la plataforma en cuestión. Los usuarios pueden conseguir estos logros. Algunas clasificaciones dependen de la consecución de estos logros y los puntos asociados a cada logro. En otras implementaciones, se sigue un sistema de experiencia y niveles, de manera que se aumenta el nivel del usuario acorde a la experiencia que va consiguiendo según los logros.

La plataforma pionera en cuanto a hospedaje de logros fue Kongregate, orientada a juegos Flash, aunque no dejaba claros los privilegios que ofrecían al usuario la consecución de los logros. En esta plataforma se podían conseguir Badges en los diferentes juegos y Puntos que permitían aumentar el nivel de los usuarios. (Kongregate FAQ, actualizado en 2015)

Las implementaciones tienen tanto aspectos similares entre ellas como aspectos característicos de una plataforma individual. Todas las implementaciones coinciden en que los usuarios no pueden repetir logros, que los logros son dependientes del juego y que la clasificación de los usuarios dependerá del conjunto de insignias que logra un usuario. Sin embargo, ninguna aplicación comparte nombre en cuanto a los logros, y las características de los mismos pueden ser algo específicas. Las veremos en el apartado individual dedicado a cada plataforma.

En el caso de Steam, los logros se denominan Insignias, en Playstation Network se conocen como Trofeos, en Xbox Live se almacenan como Achievements y en Google Play existen tanto Logros como Misiones. Dentro de cada uno de los apartados de cada plataforma se estudiará el modo en el que están definidos los logros de las mismas en mayor profundidad.

## Sección 5.2. Plataformas existentes y consolidadas.

A continuación, vamos a realizar un estudio sobre las plataformas de mayor repercusión social que incluyen tanto juegos como recompensas/logros en los mismos.

Empezaremos por la plataforma Steam, cuyos juegos están orientados a diferentes formatos en cuanto a plataformas. Posteriormente analizaremos dos plataformas orientadas a diferentes videoconsolas, como son Xbox Live, orientada a Xbox One y Xbox 360, además de juegos de sistemas operativos Microsoft y Playstation Network, orientado a las diferentes consolas desarrolladas por Sony, tanto tradicionales (PlayStation 3 y Playstation 4 ) como portátiles (Playstation Portable y Playstation Vita), pertenecientes a las últimas generaciones de videoconsolas. Para acabar, se estudiará Google Play, orientado a juegos para plataforma android.

Se estudian éstas y no otras, como puede ser la nintendo eShop, porque además de ofrecer los servicios de distribución de juegos disponen todas ellas de un sistema de logros en el que nos inspiramos y adaptamos a nuestro proyecto, como veremos en capítulos posteriores. Destacar que entre las que estudiamos, casi todas, solo a excepción de Google Play, están orientadas a juegos multiplataforma.

### Sección 5.2.1. Steam


**Figura 12: Logo Steam**

Steam es una plataforma desarrollada por Valve Corporation que permite comunicarse a los jugadores de todo el mundo y ofrece servicios a los usuarios, así como a desarrolladores tanto mayoritarios como minoritarios. El número de cuentas activas se estima en 125 millones, y dispone de un amplio catálogo de juegos, cuyo número asciende aproximadamente a 4500 (Anandtech, 2015). Los primeros juegos se encuentran disponibles en exclusiva para el sistema operativo Windows. Con el paso del tiempo Steam se adapta al mercado y dispone además de videojuegos en plataformas Linux, Mac, iOS y Android. Además, se crea un sistema operativo, SteamOS, actualmente en fase beta, basado en la distribución Debian de Linux, y que será el sistema operativo principal de la línea de videoconsolas Steam Machines, aunque podrá ser instalado sobre cualquier PC. El sistema pretende maximizar el rendimiento de los juegos.

## *Características y servicios*

- Se dispone de un sistema de comunicación entre usuarios, disponible tanto de manera textual como por voz.
- Se permite identificar los juegos a los que tus amigos o miembros de grupo están jugando, así como invitar a partidas multijugador a los mismos en el caso de que esté disponible dicho servicio para los usuarios (algunos juegos están enfocados al uso individual y no permiten el modo multijugador)
- Valve dispone de foros donde los usuarios pueden opinar o hacer comentarios referentes a la comunidad y los juegos.
- Se dispone de un servidor para la búsqueda de partidas multijugador, de manera que usuarios sin amigos dentro de la plataforma o usuarios con amigos desconectados pueden jugar contra usuarios de todo el mundo en todo momento. Además, se permite sobre estos servidores la creación de lobbies con amigos o miembros de grupos.
- Se dispone de un servicio denominado Valve Anti-Cheat o VAC que permite ver qué usuarios están haciendo trampas sobre los juegos para reportarlos y se puede imponer sanciones a los mismos.
- Los jugadores pueden añadir a su biblioteca juegos no incluidos en el catálogo de Steam de manera que pueden acceder a ellos desde la plataforma.
- Los juegos no pueden venderse una vez comprados, pero sí pueden ser regalados a otros usuarios.

## *Usuarios y Logros*

- Para poder acceder a los juegos disponibles en el catálogo de Steam, los usuarios deben estar registrados en la página. Esto permite no sólo el acceso a los juegos, sino también a toda la comunidad y sus servicios.
- Se crea para cada usuario una página propia que muestra sus grupos, sus amigos, su biblioteca de juegos obtenidos, sus juegos deseados y otras opciones referentes a la parte social. Todas estas opciones pueden ser configuradas para no ser visibles por el resto de usuarios si se considera oportuno.
- Se pueden obtener cromos para los juegos. Estas dependen del juego en cuestión y están determinados por el mismo. Una vez conseguidos todos los cromos para un juego, se conseguirá una Insignia (icono en tu perfil de Steam, asociado a tu cuenta, que representa cuántos sets completos de cromos has conseguido) (SteamCromos FAQ, 2015). Además, se podrán conseguir Insignias acorde a ciertos objetivos de la comunidad cumpliendo objetivos dentro de la misma. Se podrá visualizar los cromos y las Insignias desde el apartado con el mismo nombre una vez nos hayamos identificado en la plataforma.
- Se permite a los usuarios la identificación de amigos entre los otros usuarios. Además, se puede bloquear a otros usuarios y ver aquellos con los que has jugado recientemente. Todo esto es accesible desde la pantalla de amigos.
- Es posible formar grupos que formen parte de la Comunidad Steam de manera que varios usuarios pertenecen a dichos grupos. Un usuario puede pertenecer a diferentes grupos. Un usuario puede crear grupos o simplemente unirse a un grupo, así como dejar un grupo al que estás afiliado. Estas actividades estarán disponibles desde la pantalla de grupos.

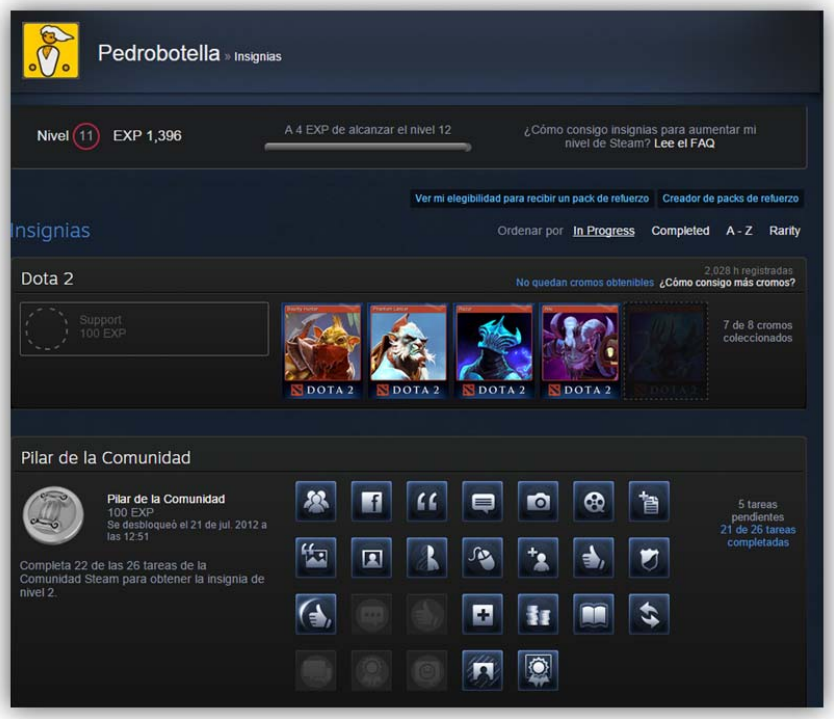



**Figura 13: Insignias Steam**


**Figura 14: Amigos Steam**

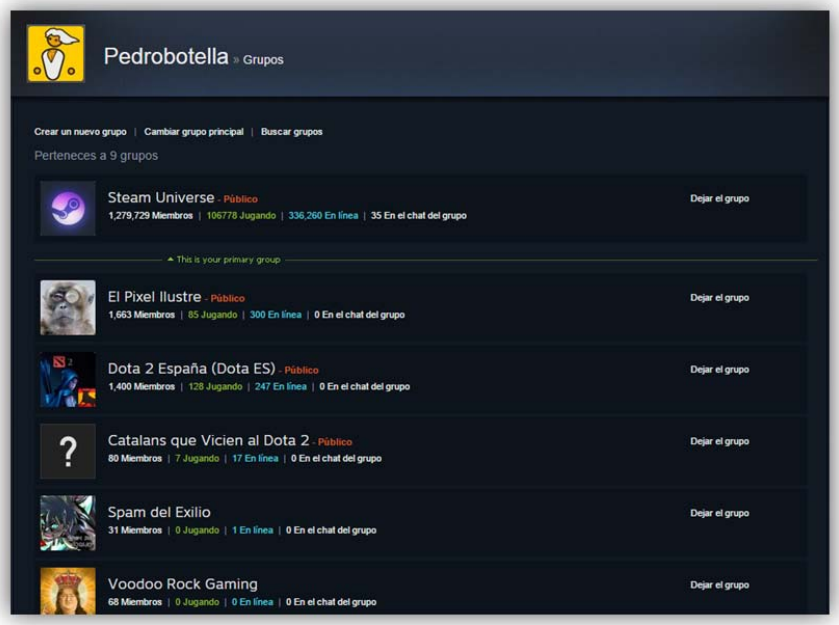



**Figura 15: Grupos Steam**

- Se dispone de una tienda en la que los usuarios pueden comprar juegos o características que mejoran la jugabilidad. Dentro de la misma se puede realizar el pago con diferentes divisas según la geolocalización del usuario.

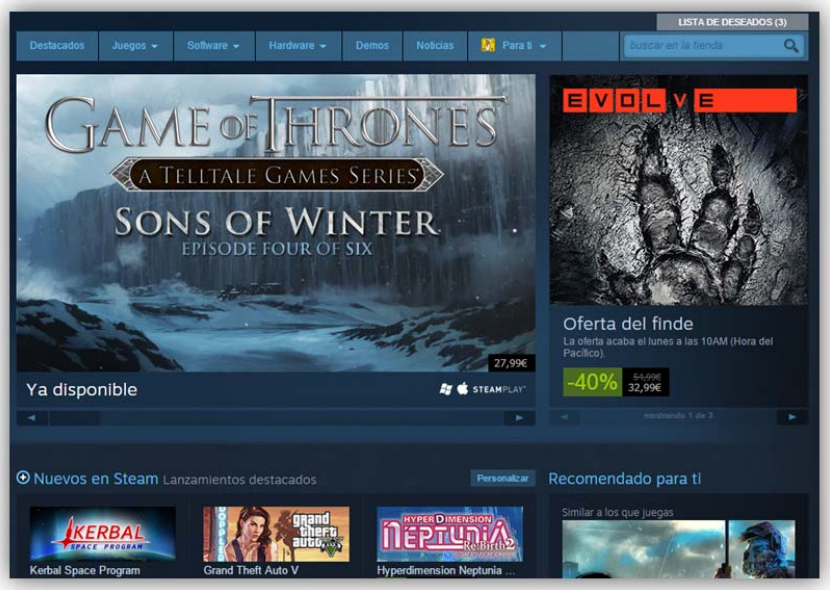



**Figura 16: Pantalla principal de Steam**

- El usuario avanza de nivel acorde a los puntos que consigue a través de las Insignias. Aparece el nivel detallado en la sección Insignias, de modo que se indicará el número de puntos que debe conseguir el usuario para llegar al siguiente nivel.

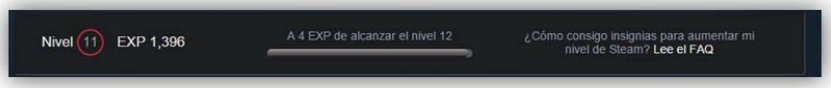



**Figura 17: Nivel Experiencia Steam**

- El perfil de un usuario permite ver ciertas características de los usuarios respecto a los juegos. También permite ver el nivel del usuario con menos detalle que la sección Insignias.


**Figura 18: Nivel de usuario Steam**

## Sección 5.2.2. Xbox Live


**Figura 19: Logo Xbox Live**

Xbox Live (abreviadamente, XBL) es el servicio de videojuegos en línea de Microsoft. Da soporte a videojuegos en Xbox One, Xbox 360 y Xbox, así como a videojuegos de sus sistemas operativos, Microsoft Windows y Microsoft Phone. Está disponible en varios idiomas y países.

Los usuarios pueden registrarse gratuitamente, de manera que obtienen un servicio denominado **"Silver"** por Microsoft. El servicio **"Gold"** ofrece ciertas ventajas a los usuarios a cambio de pago de una suscripción mensual o anual.

Gamertag es el nombre que identifica a los usuarios. Debe de ser único para cada usuario y tiene hasta 15 caracteres por nombre. Además, se prohíbe el uso de palabras internas que puedan resultar ofensivas para otros usuarios. El avatar es una caracterización virtual tridimensional y caricaturizada de los usuarios, asociada a la gamertag. Gamercard es el panel de información para analizar el perfil de usuario de la Xbox Live de Microsoft. La gamercard permite entre otras cosas acceder al Gamerscore, sistema de seguimiento y acumulación de achievements (logros del usuario)

### *Características y servicios*

- El servicio Gold permite jugar a juegos en modo multijugador, mientras que el servicio Silver permite la visualización de las características de los usuarios.
- Existe chat de voz y de video entre usuarios.
- Se dispone de un sistema de grupos temporales de hasta ocho jugadores para ver películas o jugar online a un juego.
- Se dispone de un sistema de televisión para los usuarios, independientemente del tipo de servicio asociado a la cuenta. Sin embargo, está restringido por zonas de manera que es variable en función de la geolocalización

- **(Exclusivo XBOX One)** Capacidad de grabar “gameplay” (secuencia de juego) utilizando la aplicación de Upload y su característica DVR (Digital Video Recorder). Para utilizar esta característica es necesario tener servicio Gold.
- **(Exclusivo XBOX One)** Reconocimiento por Kinect de la voz para acceder a las distintas aplicaciones
- **(Exclusivo XBOX One)** Reconocimiento facial por Kinect para acceso al sistema.
- **(Exclusivo XBOX One)** Sistema de edición de los gameplays previamente grabados. Para esto es necesario disponer del servicio Gold.

## *Usuarios y logros*

- El Gamerscore es un sistema de acumulación de puntos que refleja el número de achievements que ha acumulado un usuario de Xbox Live mostrando el número de puntos acumulados. Estos puntos dependen de los achievements conseguidos, que tienen asociados puntos. El Gamerscor refleja la suma de los puntos totals.
- Para poder acceder al apartado social, los usuarios deben estar registrados en XBox Live. Para ello deben tener asociado un gamertag. El gamertag es el identificador de cada usuario, y conforme a él se accede a su gamercard, la cual incluye avatar, gamerscore, reputación de usuario o juegos a los que el usuario ha jugado recientemente.
- Los achievements están asociados a los juegos y son limitados por los mismos. Se definen por los desarrolladores. Tienen asociados puntos de juego.

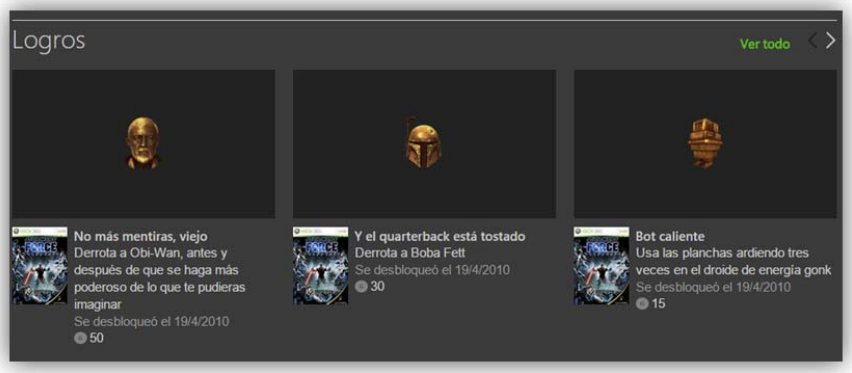



**Figura 20: Logros de usuario Xbox Live**

- El sistema TrueSkill permite a los usuarios ser recolocados para una competitividad similar entre los distintos equipos en juegos online, de manera que cada juego sea tan competitivo como sea posible. La reputación de un usuario depende de los votos que otros usuarios realizan sobre ellos al interactuar en juegos, y es visible desde su gamercard, junto a sus puntos de juego totales.

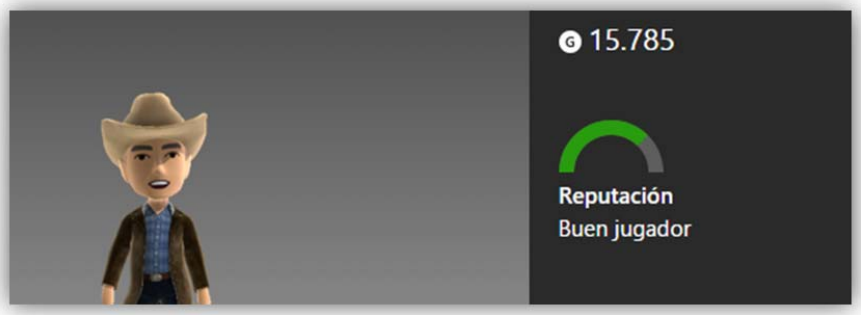



**Figura 21: Reputación de un jugador Xbox Live**

- Se pueden almacenar amigos de manera que se puede hacer intercambio de mensajes con ellos o ver sus logros.


**Figura 22: Amigos Xbox Live**

- Desde Xbox Game Store, la tienda de Microsoft, se puede acceder a videojuegos o contenido adicional de los mismos.


**Figura 23: Acceso a videojuegos y contenido adicional de Xbox Game Store**

- Los avatares pueden ser modificados. Permiten customización de ropa, peinado, género, forma del cuerpo o rasgos faciales. Se puede comprar ropa para el avatar en la tienda de Microsoft.


**Figura 24: Personalización de perfil personal en XBOX**

- Existe el concepto de imagen del jugador, también modificable, que se distingue del concepto de avatar en que es en 2 dimensiones. En la imagen inferior, que es el gamercard de un usuario, se muestra un ejemplo de Avatar (el chico con casco en 3 dimensiones) y varios ejemplos de imagen de jugador en los amigos.


**Figura 25: Imagen del jugador Xbox Live**

## Sección 5.2.3. Playstation Network

**Figura 26: Logo PlayStation Network**

Es la plataforma de servicios multijugador que pone Sony Computer Entertainment al servicio de los usuarios de sus sistemas Playstation 3, Playstation 4, PsVita y PlayStation Portable. Surge en 2006 para descarga de juegos en Playstation 3 y PSP.

### *Características y servicios*

- Creación de partidas multijugador. Para ello, a diferencia de XBOX Live, no se necesita disponer de un servicio especial, sino que todo usuario puede crearla.
- Chats de voz y de texto entre los usuarios. Permite la interacción entre los usuarios del sistema

- Intercambio de mensajes entre usuarios, permitiendo también la interacción de los usuarios del sistema pero orientado a más largo plazo.
- PlayStation Home, servicio de creación de avatares, con su propios apartamentos (que podían decorarse) su propia ropa, fue suprimido el 31 de marzo de 2015.
- **(Desde las plataformas)** Descargas que incluyen juegos multijugador, películas y series de televisión. Dentro de los juegos que se ofertan, también es posible descargas demos (versiones no finales de los juegos) de manera que el usuario sale beneficiado porque puede probar los juegos sin la necesidad de gastar dinero en ellos y los desarrolladores a su vez obtienen publicidad para sus juegos.
- **(Desde las plataformas)** Navegación por la web en navegador propio. Permite la navegación por internet desde las distintas consolas de sony.

## *Usuarios y logros*

- Se permite el registro de los usuarios, asociando un mail, con nombre de usuario único. No es necesario disponer de un dispositivo con el que jugar para poder entrar a la plataforma, accesible desde internet.
- Trofeos, obtenibles en los diferentes juegos. Se asignan al usuario y dependen de ellos, tal como los achievements de XBL o los logros de Steam. Existen trofeos de diferentes categorías: oro, plata y bronce, dispuestos por los desarrolladores de mayor a menor dificultad en su obtención. Cuando el jugador obtiene todos los logros de un juego, obtiene un nuevo trofeo de categoría platino (no incluido en juegos pequeños). Los desarrolladores tienen la obligación de incluir trofeos en sus juegos.

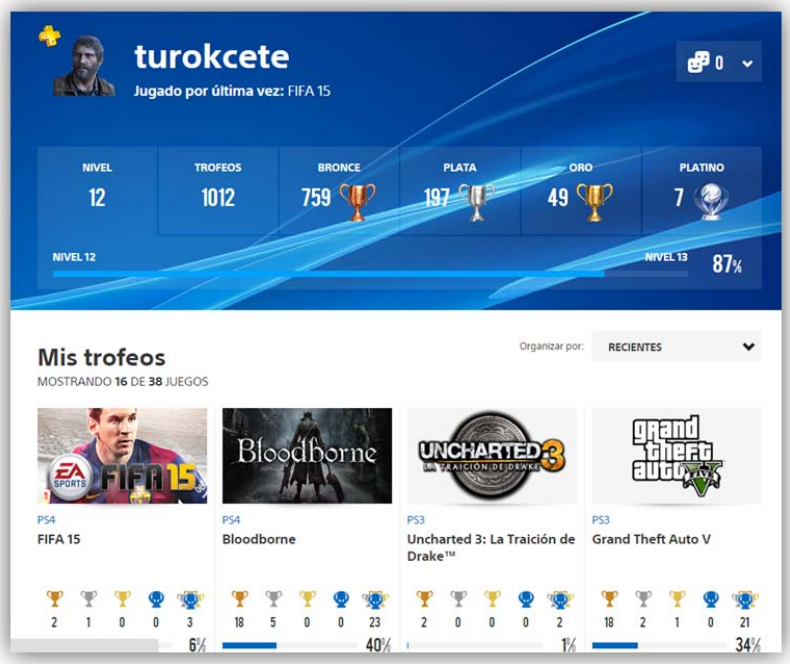



**Figura 27: Trofeos PlayStation Network**

- Rankings de usuarios, en función de los distintos trofeos obtenidos. Permite comparar tus trofeos con tus amigos.

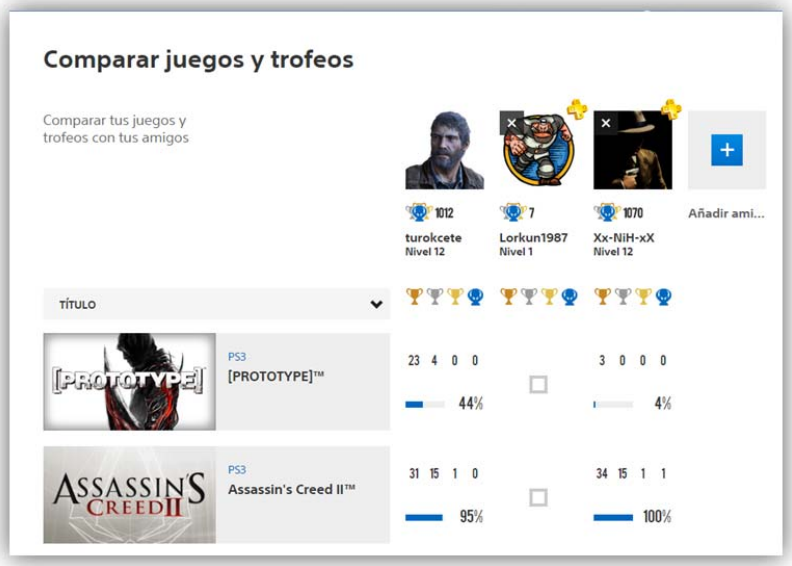



**Figura 28: Comparación de juegos y trofeos en Playstation Network**

- Listas de amigos asociados a un usuario. Cada usuario puede tener asociados amigos para competir contra ellos y realizar comparativa en cuanto a insignias conseguidas.

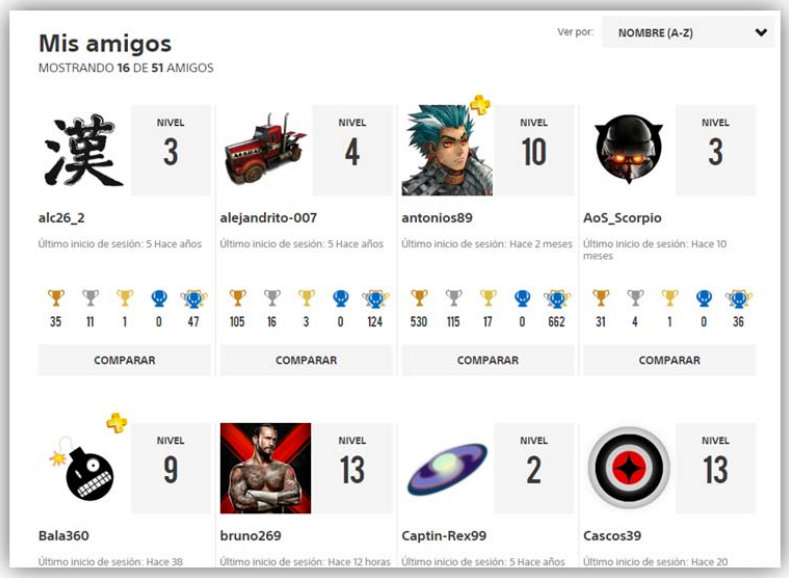



**Figura 29: Amigos PlayStation Network**

- Utilización de un avatar o imagen de perfil modificable.

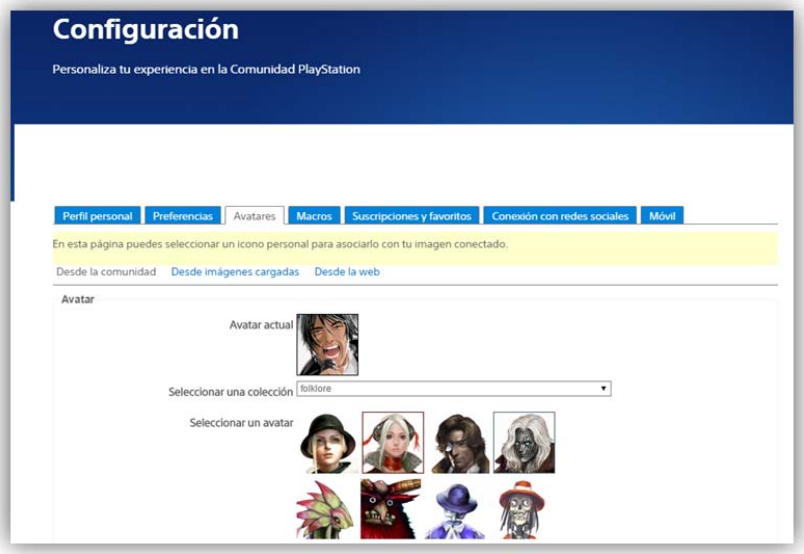



**Figura 30: Imagen o avatar PlayStation Network**

## Sección 5.2.4. Google Play Juegos


**Figura 31: Logo Google play**

Google Play es una plataforma de distribución digital para los dispositivos que operan sobre un sistema operativo Android. La distribuidora es Google. Permite a los usuarios descargar y comprar aplicaciones, juegos, música, libros, revistas y películas. Además, antaño ofrecía la venta de dispositivos móviles (ordenadores o teléfonos), exclusivos de la empresa Google Inc. Este servicio se ha suprimido con la creación de Google Store, orientado a hardware. Play Juegos es la parte orientada a juegos dentro de Google play, un servicio y aplicación desarrollado para Android, iOS y web. En este servicio se añaden las opciones de modo multijugador, logros, tabla de posiciones y de guardar información en la nube de los juegos que sean compatibles con el servicio.

## *Características y servicios*

Para hacer uso de Google Play, y de otros servicios de Google es necesario tener instalada en el dispositivo móvil la aplicación Servicios de Google Play, la cual se encarga de realizar la autentificación del usuario y otras tareas administrativas necesarias para los distintos servicios de Google. También es necesario instalar la aplicación Google Play Juegos. Esta última es la que proporciona el sistema que aquí estamos estudiando.

Las principales características de este sistema son las siguientes:

- Existe una clasificación global del usuario. Los puntos de experiencia mejoran el nivel del jugador.
- Además, existe una clasificación de experiencia por género o tipo de juego. Es una idea interesante que pone el nivel del usuario en su contexto.
  Los géneros que Play Juegos reconoce son los siguientes:

  Arcade, Estrategia, Puzle, Simulación, Acción, Aventura, Carreras, Cartas, Casino, Casual, Deportes, Educativos, Familia, Juegos de mesa, Juegos de rol, Juegos de vocabulario, Música y Preguntas y Respuestas.


**Figura 32: Distintos géneros que distingue Google Play Games**

- Existe un Historial de XP. Un *timeline* o línea de tiempo donde aparecen todas las entradas que han proporcionado experiencia al usuario en orden cronológico.


**Figura 33: Historial de XP de Google Play Games**

- Permite hacer el perfil del usuario público o privado.
- Existe una bandeja de entrada a la que puede acceder el usuario. En ella se pueden encontrar Partidas multijugador, Regalos y solicitudes, y Misiones aceptadas.


**Figura 34: Bandeja de entrada de Google Play Games**

- Play Games ofrece la posibilidad de buscar y seguir a amigos o conocidos. El concepto de *seguir* a

alguien viene impuesto por el servicio Google+, la red social con la cual está integrado el sistema (y en general, toda la plataforma de Google). Además, este sistema te recomienda a gente que quizás conozcas.

- La opción explorar te permite conocer juegos nuevos y recibir recomendaciones de otros juegos que podrían interesar al usuario, basándose en sus decisiones pasadas.

## *Usuarios y logros*

- Al jugar a un juego, se pueden obtener logros a medida que se logran ciertos objetivos. Dentro del mismo juego pueden existir diferentes logros. A diferencia de otras plataformas, este servicio es opcional y no aparece para todos los juegos.


**Figura 35: Logros de Google Play Games**

- A medida que se desbloquean los logros, se consiguen puntos de experiencia en el perfil de Play Games para el usuario.

- En algunos juegos existe una posibilidad alternativa a los logros que otorgará puntos de experiencia: las denominadas misiones. El funcionamiento es similar a los logros pero la diferencia entre éstos radica en que los retos deben superarse con un tiempo limitado en las misiones. El apartado de misiones es un apartado diferente en cada juego; una ampliación de la funcionalidad de éstos ya que la iteración de dicha misión se realizada de manera separada a la interacción usual del juego.


**Figura 36: Puntos de XP de Google Play Games**

## Sección 5.3. Resumen y adaptación al proyecto

Tras el estudio realizado de las diferentes plataformas, se observa que todas tienen en común una implantación de logros. Esto puede ser debido al impacto que tienen sobre la motivación que obtiene un usuario a la hora de jugar, ya que a medida que se consiguen las insignias, se consiguen logros personales para cada usuario. Consideramos vital la motivación para un estudiante, con lo que a la motivación al estudio que se consigue mientras se juega, se le añade una motivación extra de obtener logros personales. Estos logros personales a su vez, implican que un usuario obtiene conocimiento. Para un juego educativo, las insignias son un aliado doble para el usuario ya que consigue a la par estos dos elementos esenciales para el progreso del estudiante, ya nombrados, que son el conocimiento y la motivación.

Ningún sistema de logros es igual para dos plataformas separadas. Después del estudio de las diferentes implementaciones, adoptamos una manera similar a la implementación de Microsoft (Xbox Live) basada en asociar puntos a sus achievements. Además, decidimos implementar algunas de las características de éstas en una nueva plataforma compuesta no sólo por los videojuegos, sino a su vez por una página web de acceso a los mismos y un servidor que almacene base de datos de insignias obtenibles, usuarios y servicios. En el capítulo siguiente se hablará de las diferentes partes de la plataforma y de todos los servicios y sistemas de gestión de usuarios, grupos e insignias.

# Bloque II Descripción del sistema

# Capítulo 6. El sistema Olimpiada FDI

Una vez se ha visualizado el problema y hemos analizado las características de los sistemas implementados en los que nos vamos a inspirar, dedicaremos este capítulo a la descripción general de la solución propuesta.

La estructura de este capítulo estará formada por una descripción global del sistema a modo de introducción, para posteriormente analizar conceptualmente sus partes por separado, entre las que se encuentran la infraestructura de servicios backend, la página web orientada a usuarios y los juegos.

Antes de entrar en materia, y para entender las relaciones entre las diferentes partes del sistema, a modo de introducción, vamos a explicar inicialmente lo que nuestra plataforma entiende por insignia.

Una insignia es un logro obtenido por un jugador para un juego concreto, definida por el juego, acorde a la consecución de un objetivo u objetivos determinados. En la implementación de una insignia en nuestro sistema, una insignia tiene asociados unos puntos, que se asignan al usuario de manera que a mayor puntuación obtenida más alto se encontrará en los distintos rankings.

El nombre está inspirado en el campus virtual de la Universidad Complutense de Madrid, que permite asignar a las asignaturas insignias que un alumno puede obtener en función de los logros obtenidos en una asignatura.

Se han incorporado insignias a nuestro sistema, siguiendo los modelos de las plataformas multijugador más conocidas, con los siguientes objetivos:

- Añadir función de multijugador a la plataforma, implementando un sistema de rankings ya probado por otras plataformas de éxito contrastado.
- Aporte de jugabilidad y entretenimiento al tener objetivos claros.
- Nexo de unión con el campus virtual de la Universidad Complutense de Madrid, de manera que la obtención de insignias podría suponer obtención de insignia del campus (esto se comentará en trabajo futuro)

## Sección 6.1. Descripción general

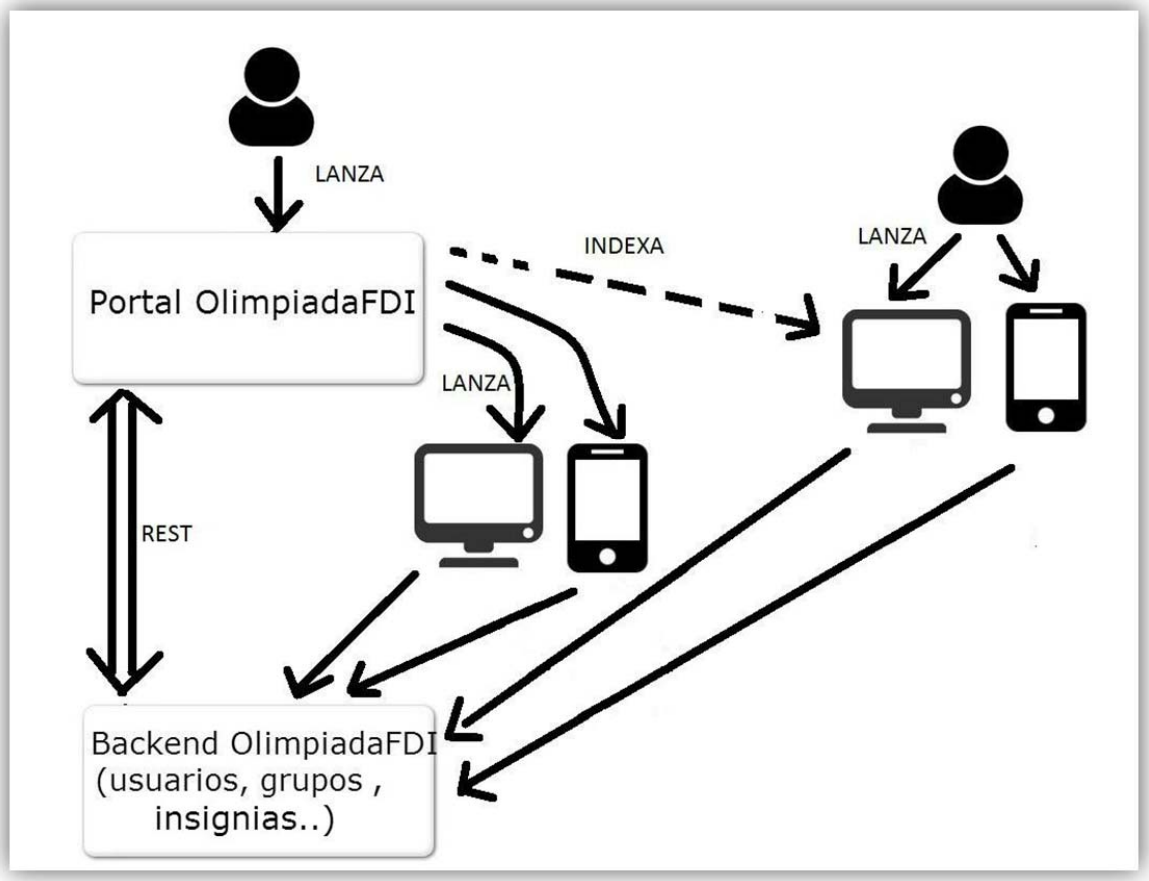



**Figura 37: Descripción general del sistema de OlimpiadaFDI**

Olimpiada FDI se compone de tres partes fundamentales:

- Un portal para encontrar los juegos de OlimpiadaFDI, ver las puntuaciones y hacer un seguimiento de las insignias de los usuarios.
- Un backend de servicios REST para la gestión de usuarios e insignias.
- Una suite de juegos a modo de implementaciones de referencia, realizados con distintas tecnologías y centrados en distintas materias.

Todos estos elementos forman una nueva plataforma multijugador para juegos educativos diversos, con servicios similares a los de las plataformas descritas en el Capítulo 5. Al igual que en dichas plataformas, el objetivo es que distintos desarrolladores puedan hacer sus propios juegos compatibles con la plataforma OlimpiadaFDI.

Así, el proyecto tiene como núcleo el backend de servicios REST. Este backend es el que gestiona las listas de usuarios, lleva la cuenta de las insignias disponibles en cada juego, gestiona la asignación de insignias a cada usuario, almacena los grupos y puntuaciones de cada usuario y, en general, gestiona la lógica de las competiciones e insignias.

Este backend ofrece servicios que están diseñados para ser usados tanto por los juegos como por el portal de OlimpiadaFDI. El portal permite a los usuarios registrarse, encontrar los juegos, formar los grupos y ver los perfiles de cada usuario con sus puntuaciones, nivel de experiencia e insignias recibidas. El portal es la forma más visible de interacción, pero todas las operaciones realizadas en el portal se registran en el backend a través de los servicios web. En la práctica, esto permitiría un modelo similar al del servicio Steam (Sección 5.2), que permite interactuar indistintamente a través de la web de Steam o de aplicaciones desarrolladas para las distintas plataformas (Windows, OSX, Linux, iOS y Android).

A su vez, el backend orientado a servicios también permite que los juegos se comuniquen con el servidor para intercambiar información de usuarios (e.g. login) y para notificar cuándo un jugador consigue una insignia concreta. Al ser una API de servicios REST abierta, los desarrolladores simplemente tendrían que hacer sus juegos compatibles con esta API para poder registrar badges en nuestro sistema.

Para ejemplificar este modelo, hemos desarrollado también un conjunto de juegos, implementados con distintas tecnologías y enfoques, para ejemplificar como nuestra infraestructura podría dar soporte a un ecosistema de juegos muy variados en distintas plataformas.

En las próximas secciones se describen más a fondo estos tres componentes principales del proyecto OlimpiadaFDI (el backend, el portal de juegos y el conjunto de juegos desarrollados como implementación de referencia).

## Sección 6.2. APIs para gestión de insignias

Para la parte de insignias hemos creado una API que nos ayuda a gestionarlas, tratarlas y mostrarlas.

Como parte de gestión tenemos 6 métodos que nos permiten:

**Tabla 8: Métodos de la API de gestión de Insignias**

| Método | Breve descripción |
|---|---|
| Insertar usuario | Inserta un usuario en base de datos (crear usuario) |
| Insertar insignia | Inserta una insignia en base de datos (crear insignia) |
| Insertar grupo | Inserta un grupo en base de datos (crear un grupo) |
| Asignar insignia a usuario | Asigna una insignia ya creada a un usuario ya creado |
| Asignar usuario a grupo | Asigna un usuario ya creado a un grupo ya creado |
| Borrar usuario de grupo | Borra un usuario perteneciente a un grupo |

Con ellos podemos tratar todo lo relacionado con la creación de usuarios, insignias y grupos y las posibles relaciones entre ellos.

Para la implementación de la web y las utilidades que ofrece existen 7 métodos que nos ofrecen las siguientes funcionalidades:

Tabla 9: Métodos para la API de las utilidades de la página web

| Método | Breve descripción |
|---|---|
| Login usuario | Permite con un usuario y su contraseña hacer login |
| Obtener insignias de usuario | Recibe un nombre de usuario y devuelve una lista con sus insignias |
| Obtener puntos de insignias de usuario | Recibe un nombre de usuario y devuelve la suma de los puntos de sus insignias |
| Obtener grupo por nombre de usuario | Recibe un nombre de usuario y devuelve el id del grupo al que pertenece |
| Obtener usuarios pertenecientes a un grupo | Recibe el id de un grupo y devuelve los nombres de los usuarios pertenecientes a el |
| Obtener todos los grupos | No recibe parámetros y devuelve todos los grupos existentes en base de datos |
| Obtener todos los usuarios | No recibe parámetros y devuelve todos los usuarios pertenecientes en base de datos |

Esto nos permite dejar acceder a un usuario a la web, y una vez dentro mostrarle toda la información que solicite relacionada con las insignias y los grupos.

Con todos estos métodos conseguimos la funcionalidad buscada en la infraestructura y en la página web. Todos ellos se encuentran disponibles a través de las urls detalladas en el anexo.

## Sección 6.3. Portal de Olimpiada FDI

Si en el apartado anterior hemos enumerado algunos de los servicios que ofrece la plataforma mediante su back-end, el usuario debe ser capaz de acceder a los mismos. Para ello, se desarrolla un front-end con el que el usuario interactúa. El usuario se puede comunicar de manera directa con un portal que a su vez se relaciona directamente con los juegos, lanzándolos; y con el back-end por medio de servicios REST.

No es condición necesaria para el acceso a los juegos que el usuario acceda al portal. Sin embargo, sí lo es para acceder a la gran mayoría de sus servicios. Para el acceso al portal, es necesario que el usuario se registre.

Dentro del portal, se puede acceder a la visualización de las insignias, a la visualización de grupos de usuarios y rankings tanto de usuarios como de grupos. Además de esto, existe la posibilidad de acceder a algunos juegos disponibles para la plataforma en la que nos encontremos. Para éstas, se efectúa una diferenciación de manera que acorde a la plataforma en la que nos encontremos, la visualización del portal será diferente, en función del tamaño de la pantalla que se esté utilizando.

## Funcionalidad detallada

En esta sección se enumeran las posibilidades que ofrece el portal

- **Registro en la aplicación:** Es necesario para el acceso al resto de los servicios el estar registrado dentro de la aplicación.


**Figura 38: Registro en la aplicación a través de la página web**

Se incluye el registro en la portada del portal, para facilitar a los usuarios el mismo.

- **Identificación de usuario:** Los usuarios se corresponden a un id. Cada usuario tiene asignada una contraseña que permite su identificación. Ésta se pide junto a su nombre de usuario desde la portada del portal. Para el inicio de la sesión, que permite el acceso a los servicios generales del sistema, se deben insertar ambos en la barra superior.

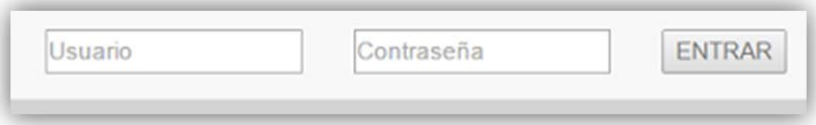



**Figura 39: Log in de la página web**

- **Acceso a juegos:** Al iniciar la sesión, en la página de inicio, se listan los juegos a los que es posible jugar (que podrían ser dependientes de la plataforma en la que nos encontremos).


**Figura 40: Acceso a juegos desde la página web**

- **Creación de grupos:** Si el usuario no pertenece a un grupo, se le permite la opción de crear su propio grupo, en el que queda insertado por defecto al instante de su creación.


**Figura 41: Creación de grupos desde la página web**

Al estar en un grupo, se modifica la barra lateral, de manera que permita al usuario salir del grupo en que se encuentra y no permita crear otro grupo (un usuario solo puede pertenecer a un grupo)


**Figura 42: Barra lateral de la web**

- **Salir del grupo:** Para este servicio se pide la confirmación del usuario


**Figura 43: Salir de un grupo desde la página web**

- **Buscar grupos:** Permite al usuario ver los grupos existentes y enrolarse al que quiera clicando sobre uno de ellos.


**Figura 44: Buscar grupos desde la página web**

- **Ver insignias obtenidas:** Permite hacer una búsqueda por juego de las insignias asociadas a un juego que el usuario tiene.


**Figura 45: Ver insignias obtenidas desde la página web**

Por supuesto, se deben haber obtenido insignias para los juegos.

- **Ver rankings individuales:** Se observan los usuarios ordenados por los puntos que han conseguido acorde a las insignias.


**Figura 46: Ver rankings individuales desde la página web**

- **Ver rankings de grupos:** Se observan los grupos ordenados por los puntos que han conseguido sus usuarios acorde a las insignias.


**Figura 47: Ver rankings de grupo desde la página web**

# Capítulo 7. Juegos desarrollados

A modo de prueba conceptual, siempre relacionados con la enseñanza de la informática, se han desarrollado juegos para diferentes plataformas. Estos juegos pueden servir como implementación de referencia a futuros desarrolladores de juegos para la plataforma.

Empezaremos definiendo los juegos que se han implementado en entorno multiplataforma con sus diferenciaciones en cuanto a la plataforma concreta, seguidos por los juegos orientados a una plataforma concreta. En el tema 8 se describe el proceso de desarrollo de los juegos aportados.

## Sección 7.1. Quicktest

***Disponibilidad:*** Aplicación web, aplicación de escritorio y Android.

***Definición:*** Quicktest es un juego cuyo objetivo consiste en responder un mayor número de preguntas posibles dentro de un determinado cuanto de tiempo, de manera que a más velocidad sea posible obtener más insignias, y que al acabar el tiempo no se pueda seguir respondiendo a las preguntas.

***Desarrollo:*** Durante 60 segundos se tratará de responder a 10 preguntas enunciadas al azar sobre 6 categorías diferentes:

- Programación
- Ingeniería del software
- Bases de datos
- Redes
- Sistemas operativos
- Computadores

La pregunta no desaparecerá hasta que hayamos contestado, sumando 1 punto en caso de acierto y dejando la puntuación igual en caso de fallo.

El reloj descenderá segundo a segundo y no dependerá de aciertos o fallos en las preguntas.Al contestar a las preguntas se avisará al jugador de si ha acertado o no la pregunta. Al finalizar, se mostrará una pantalla con las insignias que ha logrado el jugador durante su partida. Estas insignias serán también almacenadas en la base de datos en el caso de que el jugador no las tuviera.

En el caso de las preguntas, se reciben por peticiones al servidor, que devuelve de la base de datos de preguntas (definida en el capítulo correspondiente al servidor)

***Insignias obtenibles:*** En total, serán obtenibles hasta nueve insignias, de manera que en una sola partida no es posible obtener todas las insignias.

**Tabla 10: Insignias de Quicktest**

| ID | NOMBRE | DESCRIPCION | PUNTOS |
|---|---|---|---|
| 1 | REY | Todas acertadas | 300 |

| | | | |
|---|---|---|---|
| 2 | EL 7 MÁGICO | Al menos 7 aciertos | 100 |
| 3 | RÁPIDO Y EFICAZ | 5 aciertos en menos de 30 segundos | 300 |
| 4 | GENIO EN PROGRAMACIÓN | Acertar 3 preguntas del tipo 1 | 100 |
| 5 | GENIO EN BASES DE DATOS | Acertar 3 preguntas del tipo 2 | 100 |
| 6 | GENIO EN IS | Acertar 3 preguntas del tipo 3 | 100 |
| 7 | GENIO EN REDES | Acertar 3 preguntas del tipo 4 | 100 |
| 8 | GENIO EN COMPUTADORES | Acertar 3 preguntas del tipo 5 | 100 |
| 9 | GENIO EN SISTEMAS OPERATIVOS | Acertar 3 preguntas del tipo 6 | 100 |

***Ejemplos de partida:***

**Aplicación de escritorio:**

Es el primero de los tres juegos finales implementados en Java y recordemos tiene como objetivo responder correctamente a la mayor cantidad posible de preguntas relacionadas con las distintas asignaturas de nuestra carrera.

En primer lugar, nos aparecerá la primera pantalla del juego (Figura 48) que es la que se corresponde con el Login del jugador. En esta ventana tendremos dos posibles opciones:

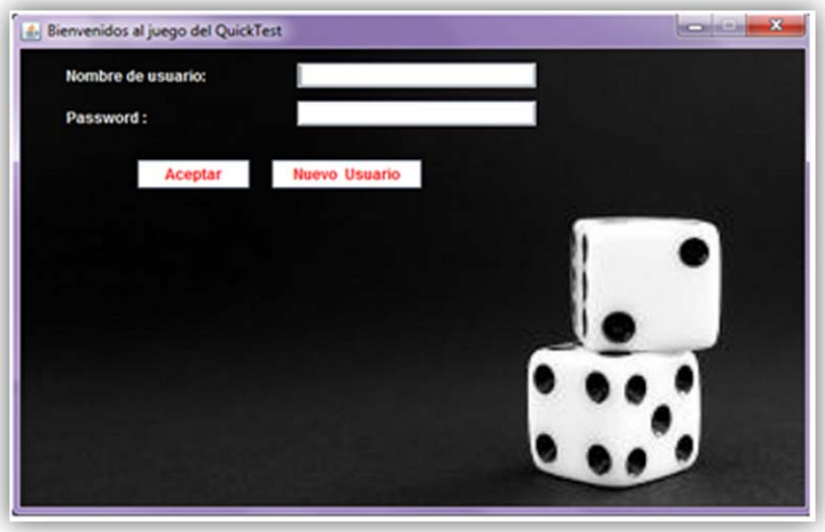



**Figura 48: Ventana inicial del juego Quicktest para aplicación de escritorio**

- Opción **Nuevo usuario**, si todavía no nos hemos registrado en el portal OlimpiadaFDI. Esta opción nos llevará a una subventana en la que tendremos un enlace para acceder directamente a la web (donde el usuario se registrará si así lo ve conveniente en ese momento) y un botón que le permitirá volver a la pantalla inicial (véase Figura 49).


**Figura 49: Ventana de Registro del juego Quicktest para aplicación de escritorio**

- Opción **Aceptar,** para confirmar los datos introducidos y dar paso al login del usuario, que en caso de ser correctos tras la verificación, nos llevará a la ventana principal de juego. En caso de que no se hubiesen introducido previamente los datos, o éstos fuesen incorrectos, se le notificará al usuario mediante una ventana emergente para posteriormente realizar un nuevo intento (Figura 50).


**Figura 50: Error de login del juego Quicktest para aplicación de escritorio**

A continuación, nos aparecerá la ventana principal en la que se desarrollará todo el juego (Figura 51). Consta de tres partes:

- **Leyenda** (parte izquierda), donde nos aparecerá cuáles son los seis distintos tipos de preguntas posibles.
- **Timer** (parte superior derecha) con una cuenta atrás que comienza con un valor igual a 60 segundos.
- **Panel** (parte inferior derecha) donde aparecerán las sucesivas preguntas con sus correspondientes respuestas.

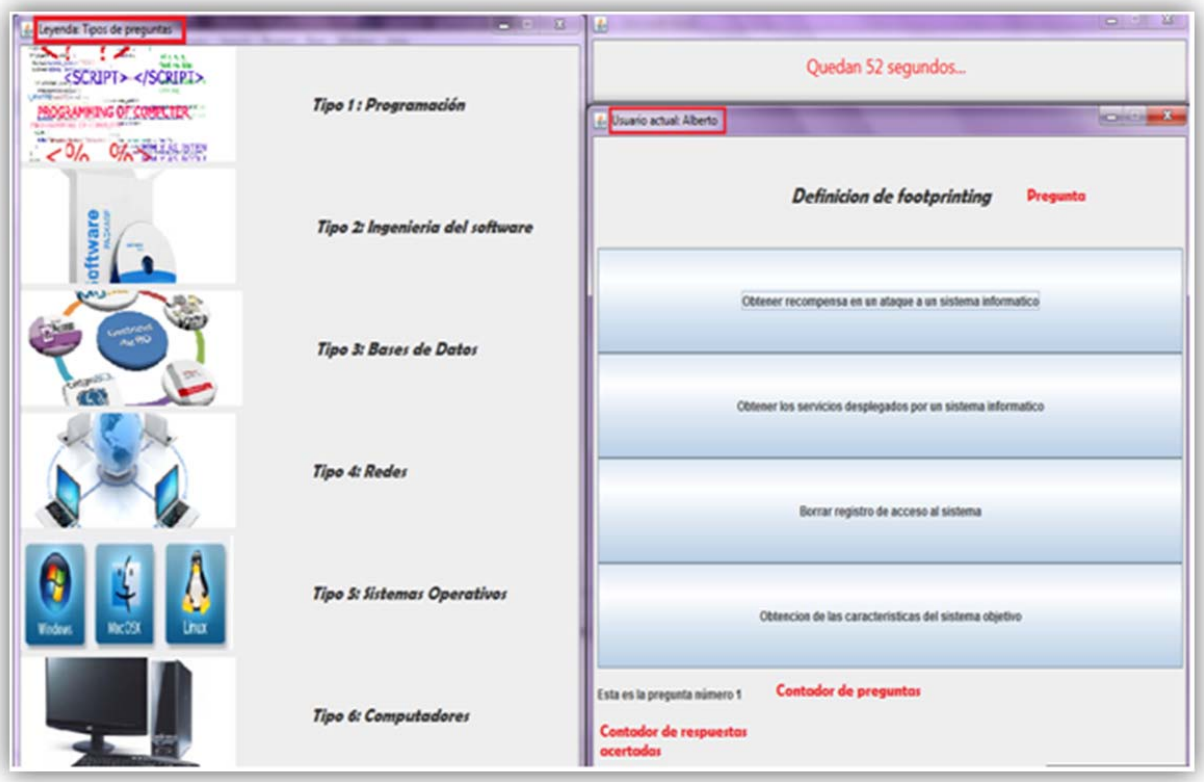



**Figura 51: Ventana principal del juego Quicktest para aplicación de escritorio**

El usuario pulsará la respuesta que crea correcta en cada caso. No aparecerá una nueva pregunta hasta que se responda correctamente a la pregunta actual. Si el usuario consigue acertar en su primer intento se contabilizará un acierto no siendo así si se requiere de más intentos. El juego continuará su curso hasta que la cuenta atrás llegue a su fin o hayan aparecido y se hayan contestado un máximo de 10 preguntas.

Finalmente, como vemos en la Figura 52, el usuario podrá visualizar un pequeño resumen de las insignias que ha obtenido en la partida.

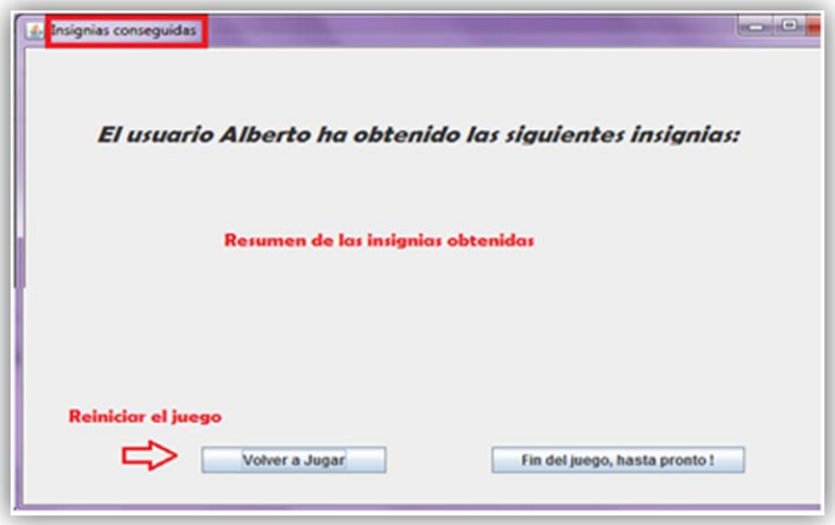



**Figura 52: Feedback de las insignias obtenidas en el juego Quicktest para aplicación de escritorio**

**Android:**

Al comenzar la aplicación, el usuario visualizará la pantalla (en Android se llaman Activities) de login, donde podrá insertar su nombre de cuenta y contraseña.


**Figura 53: Inicio QuickTest en Android**

En esa misma pantalla, si se activa el recuadro de “Recordar los detalles de mi cuenta”, se guardarán los datos de la cuenta (nick y contraseña) en la memoria interna del teléfono; de lo contrario, se borrarán.

Tras esto, se mostrará al usuario el menú principal, desde donde podrá comenzar una nueva partida, revisar la puntuación, o salir de la aplicación.


**Figura 54: Menú QuickTest en Android**

La última opción es trivial, la aplicación se cierra tras un breve mensaje de despedida. La segunda opción, la de puntuación, mostrará la información de la última partida, y cargará las insignias que tenga desbloqueadas en ese momento el jugador.


**Figura 55: Puntuación última partida QuickTest en Android**

La opción de comenzar partida iniciará un nuevo juego.


**Figura 56: Partida QuickTest en Android**

El proceso a seguir es simple, las preguntas se irán sucediendo una tras otra, hasta un máximo de 10 o hasta que el tiempo llegue a su límite. Tras cada respuesta, se comunicará al usuario si se ha respondido correctamente o no. Si se cumplen las condiciones, y se desbloquea una insignia, también se comunicará al usuario en ese instante. Tras finalizar la partida se muestra al usuario la pantalla de puntuaciones.

**Aplicación Web:**

Al clicar en la web sobre quicktest, comienza el contador a bajar desde los 60 segundos y las preguntas a aparecer. Cabe resaltar que no necesita de un log in dado que se mantiene la sesión iniciada en el portal. En todo momento se puede observar la puntuación obtenida, así como el tiempo restante. Cada pregunta tiene cuatro respuestas posibles. La pregunta se diferencia de las respuestas por el color de fondo azul, por el amarillo de las respuestas.

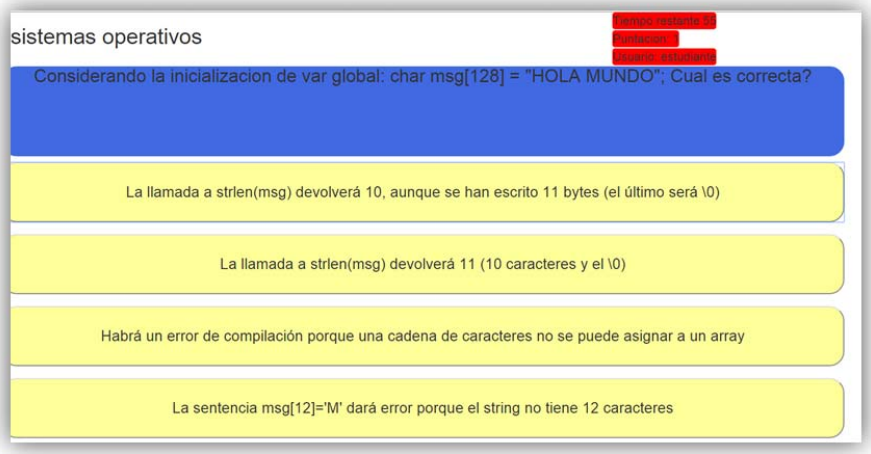



**Figura 57: Vista de las preguntas en aplicación Web**

Tras finalizar una partida, obtenemos un resumen que incluye, además del id del usuario, las respuestas acertadas de cada categoría, el tiempo restante, la puntuación obtenida y las insignias obtenidas en la iteración. Además, tiene enlaces para volver a jugar una partida o volver a la página principal.

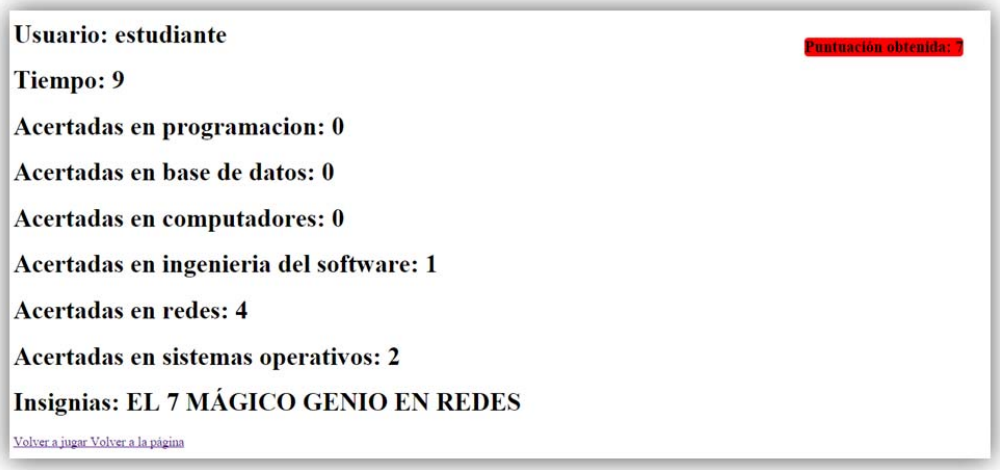



**Figura 58: Resultado QuickTest en aplicación web**

## Sección 7.2. Trivial

***Disponibilidad:*** Aplicación de escritorio y aplicación web.

***Definición:*** El juego trata de responder un mayor número de preguntas posibles dentro de un determinado cuanto de tiempo. En este caso no será como en un trivial convencional en el que compites contra otro jugador sino que ahora consiste en enfrentarse uno mismo a una serie de preguntas a contrarreloj.

***Desarrollo del juego:***

Intentaremos simular un tablero con distintas casillas de seis tipos de preguntas diferentes, otras casillas de repetición de lanzamiento del dado y una especial, la casilla central. Los saltos entre casillas dependerán del lanzamiento de un dado. El movimiento del dado desencadenará un tipo de pregunta u otra relacionada con asignaturas de nuestra carrera de seis categorías diferentes, que en principio serán:

- Programación (marrón)
- Ingeniería del software (amarillo)
- Bases de datos (azul)
- Redes (naranja)
- Sistemas operativos (rosa)
- Computadores (verde)

Una vez el juego nos ha avisado sobre qué asignatura está relacionada la pregunta, ésta aparecerá y no desaparecerá hasta que hayamos contestado. Independientemente de si la respuesta ha sido correcta o errónea, el proceso que se seguirá será siempre el mismo.

Se llevará el cuanto de un reloj que inicialmente tendrá como tiempo máximo 180 segundos, decrementando continuamente. No se harán bonus en el juego de manera que aumente o disminuya la cantidad de segundos.

Al contestar a las preguntas. se avisará al jugador de si ha acertado o no por la respuesta marcada, volviendo a lanzar el dado independientemente del acierto o fallo de la misma.

En caso de acertar, se llevará una puntuación (la cual obviamente al inicio de la partida será 0) que aumentará siguiendo las siguientes condiciones:

- Pregunta normal : suma 1 punto
- Pregunta de queso: suma 2 puntos
- Pregunta central: suma 3 puntos (esto es debido a que la pregunta central no tiene categoría y puede ser de cualquier tipo de asignatura)

Al finalizar, se mostrará una pantalla con las insignias que ha logrado el jugador durante su partida. Estas insignias serán también almacenadas en la base de datos en el caso de que el jugador no las tuviera.

***Insignias obtenibles***

El número de cada insignia representa a su id en la base de datos de insignias (definida en el capítulo del servidor)

**Tabla 11: Insignias Trivial**

| ID | NOMBRE | DESCRIPCION | PUNTOS |
|---|---|---|---|
| 101 | PREMIO! | Ganar un queso | 50 |
| 102 | VAYA RACHA | 5 preguntas seguidas | 200 |
| 103 | AMO Y SEÑOR | 10 preguntas seguidas | 500 |
| 104 | AL AZAR | acierta la casilla del centro | 50 |
| 105 | RATÓN | Consigue todos los quesos | 300 |
| 106 | PUNTOS | Conseguir más de 30 puntos | 100 |

***Ejemplos de partida:***

**Aplicación de escritorio:**

El juego del Trivial fue el segundo juego que implementamos en esta tecnología. La parte que se refiere tanto al login como a la redirección a la página web para registrar un nuevo usuario, funciona de igual manera que en los demás juegos implementados en Java pero tiene un aspecto diferente (véase Figura 59).

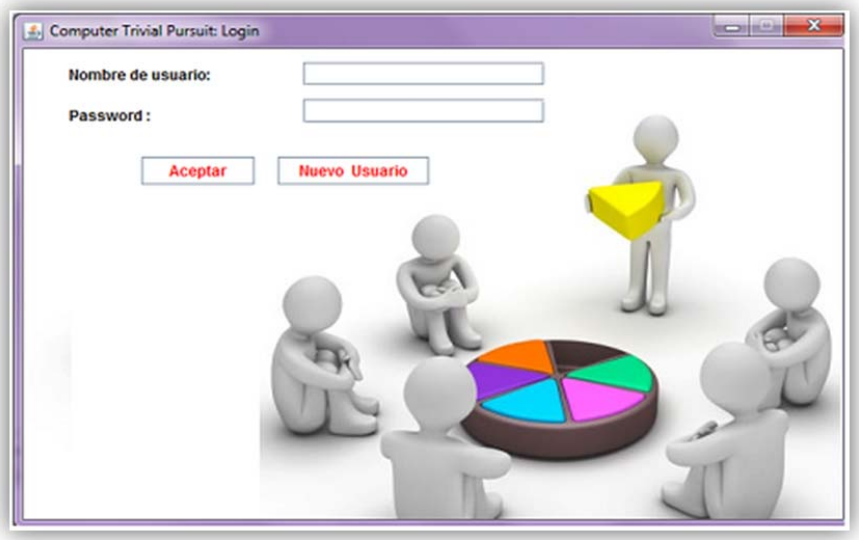



**Figura 59: Ventana de Login del juego Trivial para aplicación de escritorio**

Antes de comenzar el juego, hay una ventana inicial en la que tendremos una pequeña presentación visual del juego y en la que decidiremos si salir o empezar definitivamente la partida.


**Figura 60: Ventana inicial del juego Trivial para aplicación de escritorio**

La ventana principal de juego, al igual que en el QuickTest, se compone de tres subpaneles (Figura 61). Cada uno de ellos tiene la siguiente finalidad:

- **Leyenda** (parte izquierda), que en este caso además de aparecernos cuáles son los seis distintos tipos de preguntas y el color de la casilla con el que están relacionados, tendremos una imagen estática del tablero con cada una de sus casillas marcadas, tanto si son casillas normales (con su respectiva numeración) como si son casillas con premio.
- **Timer** (parte superior derecha) con una cuenta atrás que comienza con un valor de 180 segundos y que comenzará en el momento en el que tiremos los dados por primera vez. Inicialmente, podremos visualizar toda la ventana de juego y éste no comenzará hasta que así lo indiquemos (Ventana emergente para poder tirar los dados).
- **Panel de preguntas** (parte inferior derecha) donde aparecerán las sucesivas preguntas con sus correspondientes respuestas. Inclusive, tendremos en todo momento un feedback de cuál es nuestra puntuación actual y el número tanto de preguntas como de respuestas acertadas.

Automáticamente después de tirar los dados, el juego comenzará y se actualizarán todos los datos pertinentes. La continuación del juego quedará a la espera de que el usuario elija una posible futura casilla destino gracias a un pequeño comboBox emergente.


**Figura 61: Ventana principal del juego Trivial para aplicación de escritorio I**

Una vez realizada dicha elección, aparecerá en el panel inferior una pregunta del tipo que se corresponde con la casilla a la que nos hemos movido. En caso de acertar la pregunta se actualizarán también los marcadores oportunos (si la pregunta tenía premio, veremos que nuestro casillero de quesos se incrementa) y al

contrario que en el juego anterior, si la respuesta elegida ha sido errónea se continuará con el bucle del juego sin contabilizar ni decrementar los puntos (Figura 62).


**Figura 62: Ventana principal del juego Trivial para aplicación de escritorio II**

Por último, cuando el tiempo finalice tendremos la opción de contestar únicamente a una pregunta más. Inmediatamente después, el juego finalizará y se mostrará una ventana con el pequeño resumen de Insignias obtenidas durante la partida.

**Aplicación Web:**

Se comenzará en el tablero con un cuanto de tiempo de 200 segundos. Para avanzar, debe clicarse en el dado.

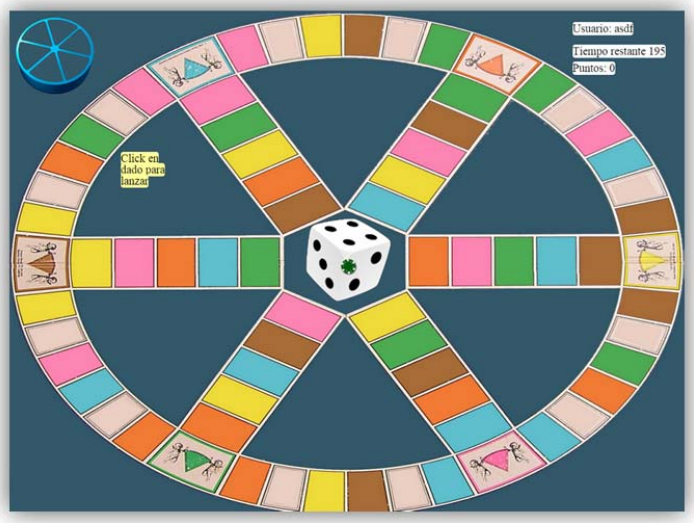



**Figura 63: Trivial aplicación Web**

Al clicar sobre el dado se mostrarán con estrellas sobre la pantalla las casillas accesibles, así como el resultado del dado

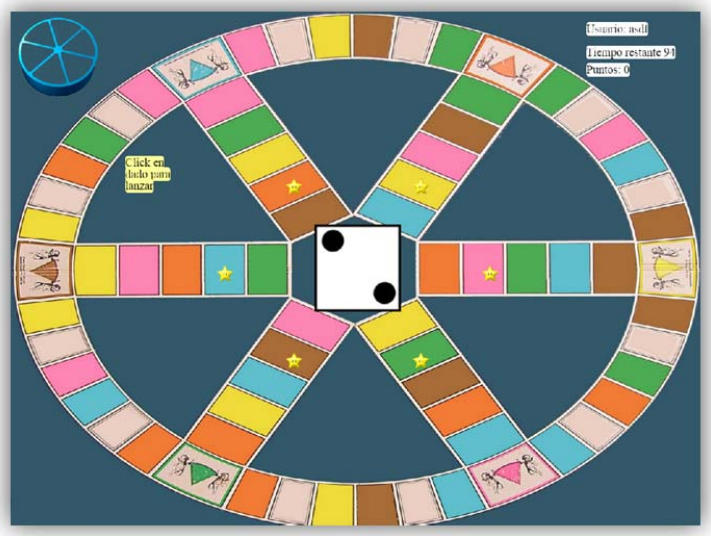



**Figura 64: Trivial aplicación Web**

En todo momento se lleva un seguimiento de puntos obtenidos y tiempo restante, así como de los quesos obtenidos. Al clicar sobre una estrella se redireccionará a una pregunta de la base de datos de preguntas. Dentro de la pregunta se podrá seguir visualizando el tiempo restante, así como la puntuación obtenida hasta el momento.

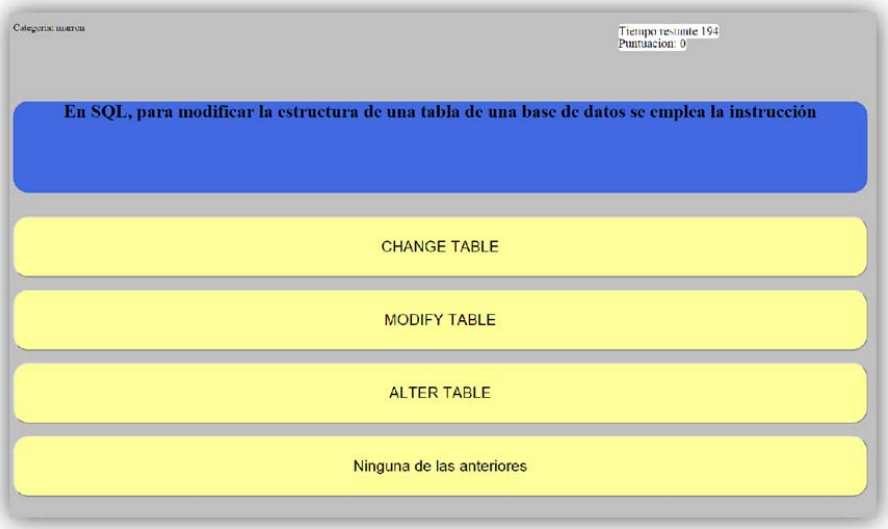



**Figura 65: Trivial aplicación Web**

Se avisará tanto si la respuesta es correcta como incorrecta por medio de un alert. En el caso de conseguir un queso en una de las casillas correspondientes (implementado de la misma manera que en el trivial tradicional) se reflejará en la esquina superior izquierda.


**Figura 66: Trivial aplicación Web**

La iteración termina cuando el tiempo llega a cero. En ese momento, se conforma un resumen componente de insignias obtenidas, tiempo restante y quesos obtenidos, así como la opción de jugar de nuevo y de volver a la página principal.

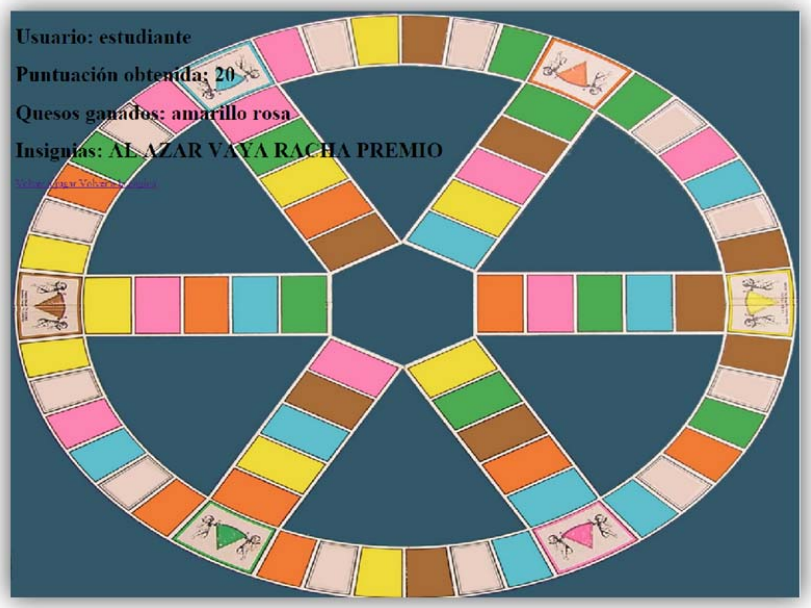



**Figura 67: Trivial aplicación Web**

## Sección 7.3. CodeChallenge

***Disponibilidad:*** Android.

***Definición:*** En este juego el usuario deberá revisar y corregir las líneas de código de un algoritmo determinado al azar. Con este juego se pretende reforzar la lógica del jugador y su capacidad para entender e interpretar el código ajeno. Para mayor presión, se establece un tiempo máximo de 60 segundos.

***Desarrollo:*** Se muestra un código al usuario con algunos errores de implementación. El jugador dispondrá de 60 segundos para identificar esos errores y marcar las líneas correspondientes;.

En el caso de marcar una línea como fallo cuando no lo es, se penalizará debidamente.

El reloj descenderá segundo a segundo y no dependerá de ninguna de las acciones del jugador. Al finalizar la partida, una vez encontrados todos los fallos o cuando el tiempo se acaba, se mostrará una pantalla con las insignias que tiene desbloqueadas el jugador, y datos de la última partida que se guardan en la memoria interna del dispositivo. Las insignias son almacenadas en la Base de Datos

***Insignias obtenibles:*** En total, serán obtenibles hasta cinco insignias, de manera que en una sola partida no sea posible obtenerlas todas.

Tabla 12: Insignias para CodeChallenge

| ID | NOMBRE | DESCRIPCION | PUNTOS |
|---|---|---|---|
| 81 | NEWBIE | Has encontrado un fallo | 100 |
| 82 | HAWKEYE | Has encontrado todos los fallos sin equivocarte | 200 |
| 83 | CRYSIS' LEAD PROGRAMMER | Te has equivocado 30 veces en una sola partida | 100 |
| 84 | QUICK REACTION FORCE | Termina la partida antes de 10 segundos | 300 |
| 85 | PRESS 'N' PRAY | Has encontrado un error a los 2 segundos de comenzar la partida | 100 |

***Ejemplos de partida:***

Al comenzar la aplicación, el usuario visualizará la pantalla (Activity) de login, donde podrá insertar su nombre de cuenta y contraseña.


Figura 68: Pantalla de login de CodeChallenge

En esa misma pantalla, si se activa el recuadro de “Recordar los detalles de mi cuenta”, se guardarán los datos de la cuenta (nick y contraseña) en la memoria interna del teléfono; de lo contrario, se borrarán.

Tras esto, se mostrará al usuario el menú principal, desde donde podrá comenzar una nueva partida, revisar la puntuación, o salir de la aplicación.


**Figura 69: Menú principal de CodeChallenge**

La última opción cierra la aplicación. La segunda opción, la de puntuación, mostrará la información de la última partida, y cargará las insignias que tenga desbloqueadas en ese momento el jugador.

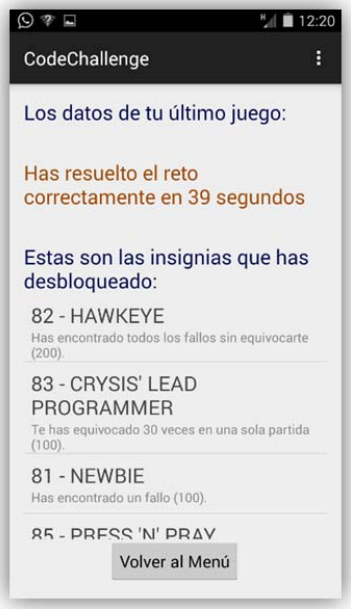



**Figura 70: Pantalla de puntuaciones de CodeChallenge**

La opción de comenzar partida iniciará un nuevo juego.

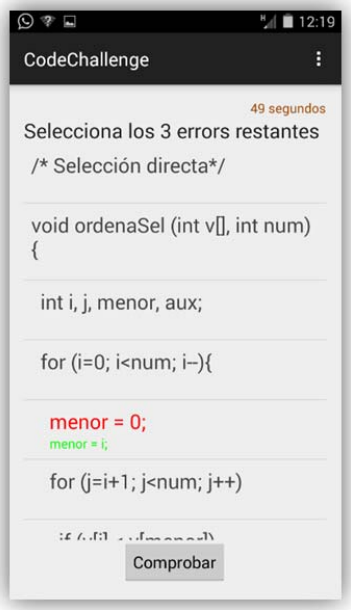



**Figura 71: Pantalla de juego de CodeChallenge**

El proceso a seguir es el siguiente: Se muestra al usuario un fragmento de código en un pseudolenguaje parecido al que se enseña en la asignatura Estructura de Datos y Algoritmos; y el usuario deberá encontrar los fallos del mismo, seleccionando las líneas que crea erróneas. Cuando crea haber terminado (los errores totales se muestran en el enunciado), pulsa el botón comprobar. Si se cumplen las condiciones, y se desbloquea una insignia, también se comunicará al usuario en ese mismo instante.

Tras finalizar la partida se muestra al usuario la pantalla de puntuaciones.

## Sección 7.4. Hacker boy

***Disponibilidad:*** aplicación web

***Definición:*** El juego trata de aprender a mantener la seguridad de sistemas informáticos de forma teórica, y aprender a diferenciar entre las distintas amenazas. El usuario a través de niveles con diferente dificultad irá recorriendo una red. Se irán obteniendo puntos a medida que avanza la iteración del juego, y con respecto a los mismos y a diferentes objetivos se irán obteniendo insignias.

***Desarrollo del juego:*** El jugador comenzará en una red que, inicialmente, no tiene amenaza alguna colocada. Toda vez que comienza la iteración alrededor de las casillas, las amenazas pueden aparecer al azar en turnos determinados. En este periodo se permitirá comprar objetos y colocarlos en distintos apartados de la red. Se podrá avanzar entre pantallas para colocar los distintos objetos comprados en las diferentes casillas del juego. Los objetos comprados estarán limitados por el presupuesto inicial.
El mapa estará compuesto por N casillas, siendo N el número de subredes que tendremos que proteger. Se podrá saltar entre subredes, mediante flechas (criterio escogido debido a lo intuitivo que puede ser y por ser un clásico método de movimiento en videojuegos de plataformas). En todo momento se podrá acceder a un mapa que mostrará todas las subredes y en cual nos encontramos.

Dentro de cada una de las subredes podremos llegar a ver la existencia o ausencia de amenazas de la subred individual en que nos encontramos. Estas amenazas pueden aparecer al azar.

Dependiendo de las mismas, se efectuarán medidas para eliminarlas. La localización de amenazas permitirá al usuario aprender el concepto de log de un sistema.

A medida que se eliminen amenazas, nuestro jefe de empresa nos entregará dinero que podremos utilizar para comprar nuevos elementos de defensa.

Los puntos se podrán conseguir:

- Comprando elementos de defensa mediante dinero (éste sólo será obtenible mediante retribución del jefe)
- Eliminando amenazas.

Según se vayan consiguiendo diferentes objetivos, se obtendrán insignias, que serán las que posteriormente permitan sumar puntos y aumentar el ranking.

Las amenazas aparecerán en mayor o menor medida en función del nivel en el que nos encontremos.

Se llevará un contador de manera que cuando éste llegue a cero acabará la partida.

Características específicas del nivel fácil:

- No se distingue entre tipos de amenazas. Se pueden encontrar gusanos o troyanos pero no se diferencian.
- Se dispone de una tienda que permite comprar firewall o antivirus de distintos niveles.
- Los firewall podrán bloquear el malware entrante.
- Los antivirus podrán detectar gusanos y/o troyanos.
- Se genera malware cada 5 movimientos.
- Logs ven si existe malware aunque no se haya informado desde el antivirus
- Las redes serán de 3x3, y disponible un mapa en el que se observa dónde estás en todo momento.

Características específicas del nivel medio:

- Se distingue entre tipos de amenazas. Se pueden encontrar gusanos o troyanos y se diferencian en que los gusanos pueden autoreplicarse en otras máquinas.
- Se dispone de una tienda que permite comprar firewall o antivirus de distintos niveles.
- Los firewall podrán bloquear el malware entrante.
- Los antivirus podrán detectar gusanos y/o troyanos.
- Logs solo informan de malware si se ha generado la información por medio de antivirus.
- Se podrá hacer una monitorización manual de manera que se muestren los gusanos y troyanos.
- Se genera malware cada 4 movimientos.
- Las redes serán de 4x4, y disponible un mapa en el que se observa dónde estás en todo momento.

Características específicas del nivel difícil:

- Se distingue entre tipos de amenazas. Se pueden encontrar gusanos o troyanos y se diferencian en que los gusanos pueden autoreplicarse en otras máquinas. Además, mayor número de amenazas. Ataques smurf, DoS al menos, escaneo de puertos?.(por ver implementación final)
- Se generarán informes de incidencias a medida que se avanza en el juego.
- Se dispone de una tienda que permite comprar firewall o antivirus de distintos niveles.
- Los firewall podrán bloquear el malware entrante.
- Los antivirus podrán detectar gusanos y/o troyanos, con mucha menor eficiencia que en otros niveles.
- Se genera malware cada 3 movimientos.
- Las redes serán de 5x5, y disponible un mapa en el que se observa donde estas en todo momento.

***Insignias obtenibles:***

**Tabla 13: Insignias para Hacker Boy**

| ID | NOMBRE | DESCRIPCION | PUNTOS |
|---|---|---|---|
| 201 | NOOB HACKER | Consigue más de 300 puntos en nivel fácil | 50 |
| 202 | EMPEZANDO A HACKEAR | Consigue más de 500 puntos en nivel medio | 100 |

| | | | |
|---|---|---|---|
| 203 | NEO | Consigue 1000 puntos en nivel difícil | 200 |
| 204 | CAZA AL PITUFO | Consigue detener a un pitufo en un ataque smurf | 50 |
| 205 | PISA GUSANOS | Desactiva un virus que ya te han metido | 100 |
| 206 | THE WALL | Consigue que tus cortafuegos bloquee tráfico no deseado | 100 |
| 207 | INFILTRADO | Desactiva algún troyano | 150 |
| 208 | TUS PRIMEROS PASOS | Mira en logs, instaura cortafuegos | 100 |

***Ejemplo de partida:***

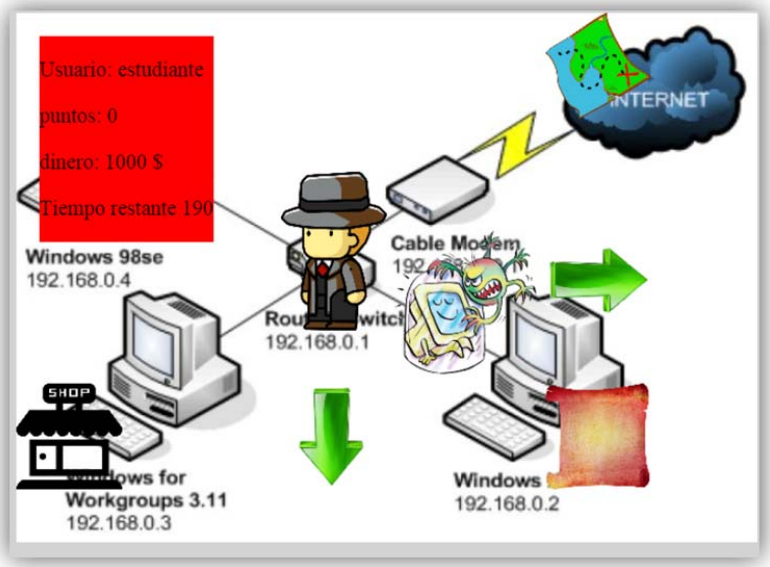



**Figura 72: Hacker Boy (1)**

Se comienza en la casilla 1-1 respecto al diferente número de casillas. A medida que el jugador clica sobre las flechas verdes, avanza por las distintas casillas. Solo aparecerán las flechas de las casillas accesibles.

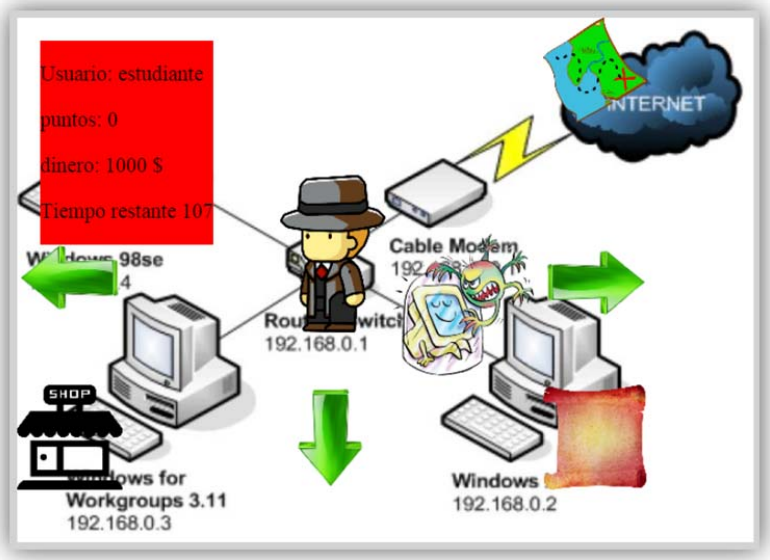



**Figura 73: Hacker Boy (2)**

Si el usuario necesita localizarse dentro del juego, puede consultar en todo momento la posición en la que se encuentra pulsando en el mapa (imagen situada en la esquina superior derecha)

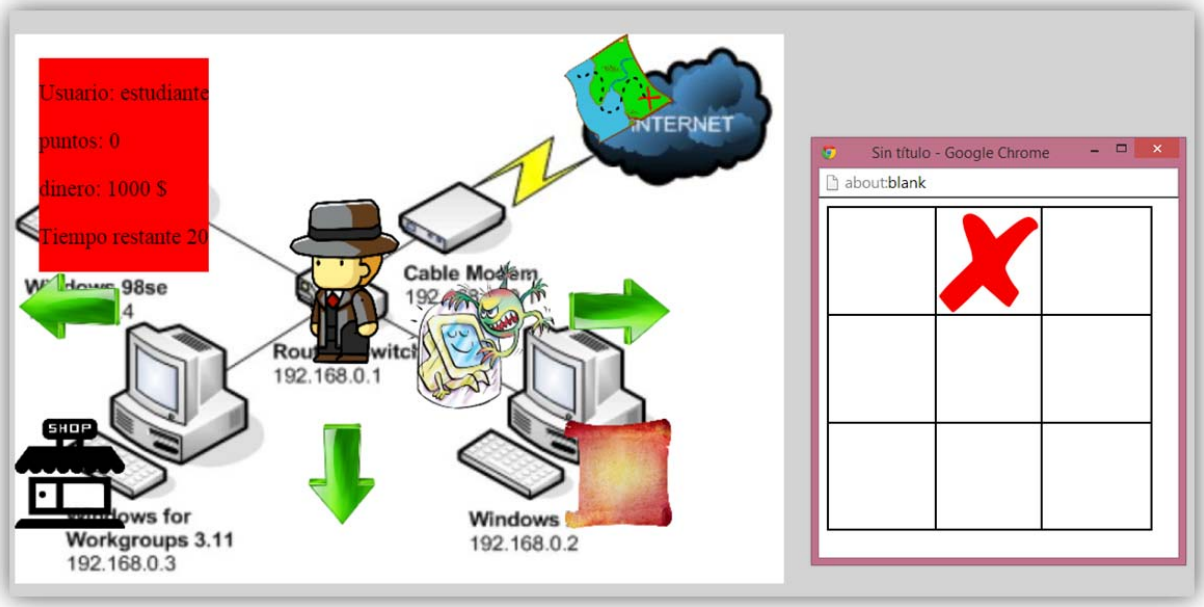



**Figura 74: Hacker Boy (3)**

Clicando sobre el botón de la tienda (esquina inferior izquierda) el usuario puede acceder a la pantalla de tienda y comprar instrumentos de defensa frente a posibles amenazas. El usuario debe pagar por ellas con el dinero que tiene acorde a los precios establecidos en la tienda.

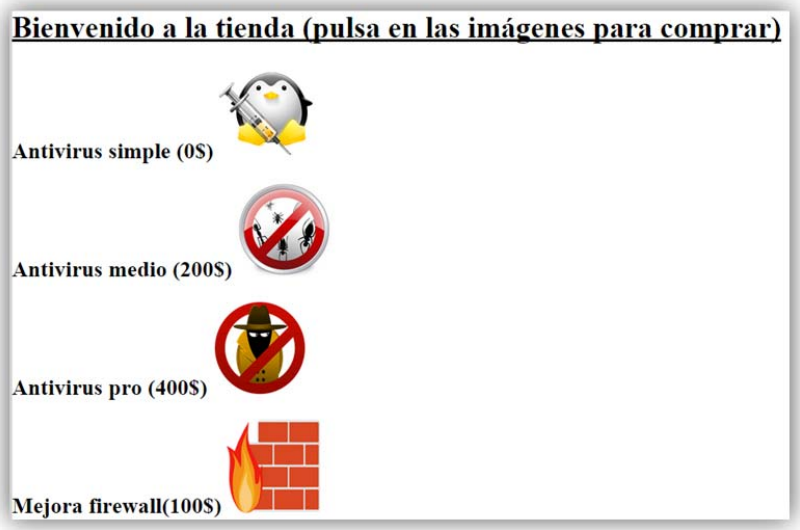



**Figura 75: Hacker Boy (4)**

La compra se asocia a la casilla en la que se encuentre.

Pulsando el botón log (botón inferior derecha) se pueden observar las amenazas dentro de la casilla localizada.

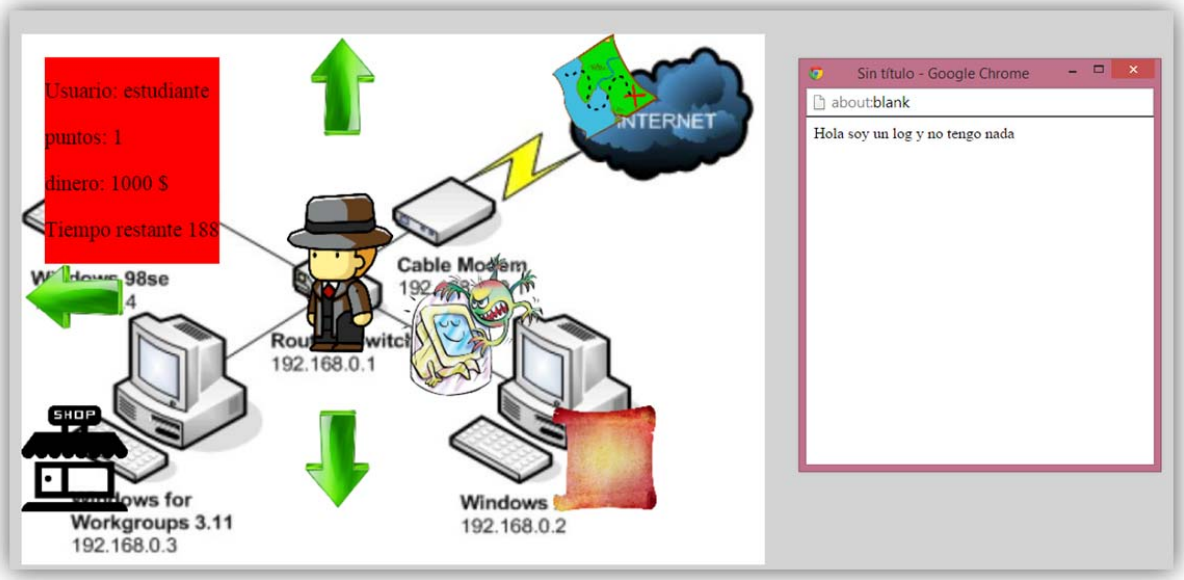



**Figura 76: Hacker Boy (5)**

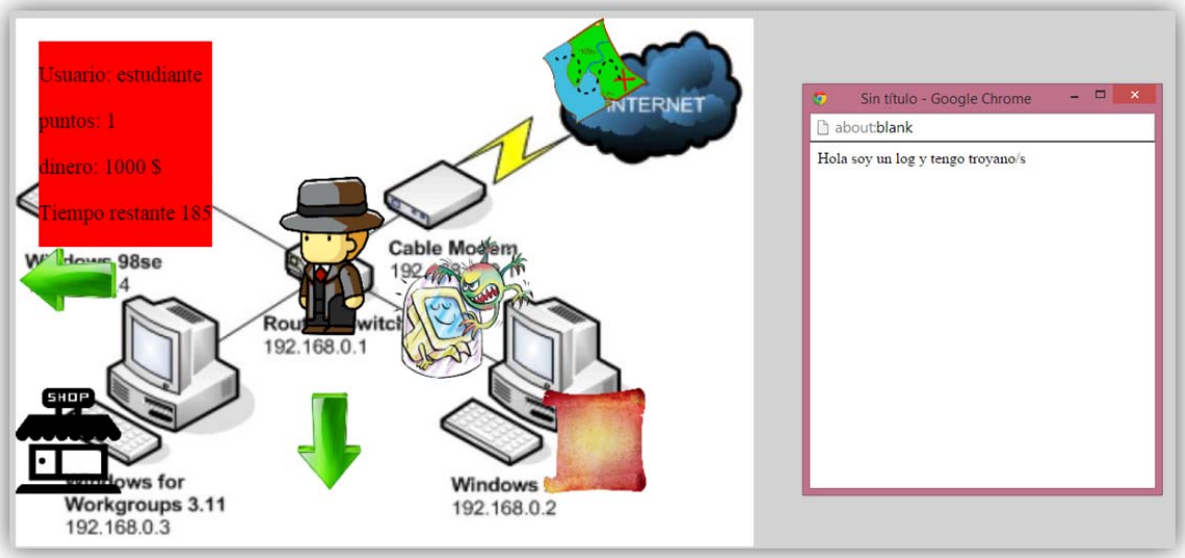



**Figura 77: Hacker Boy (6)**

En el caso de pulsar el botón arreglar amenazas, situado a la derecha de la imagen del investigador central de la pantalla, existen dos opciones: Caso de que exista una amenaza el usuario elimina la amenaza, con un mensaje indicativo de la situación.

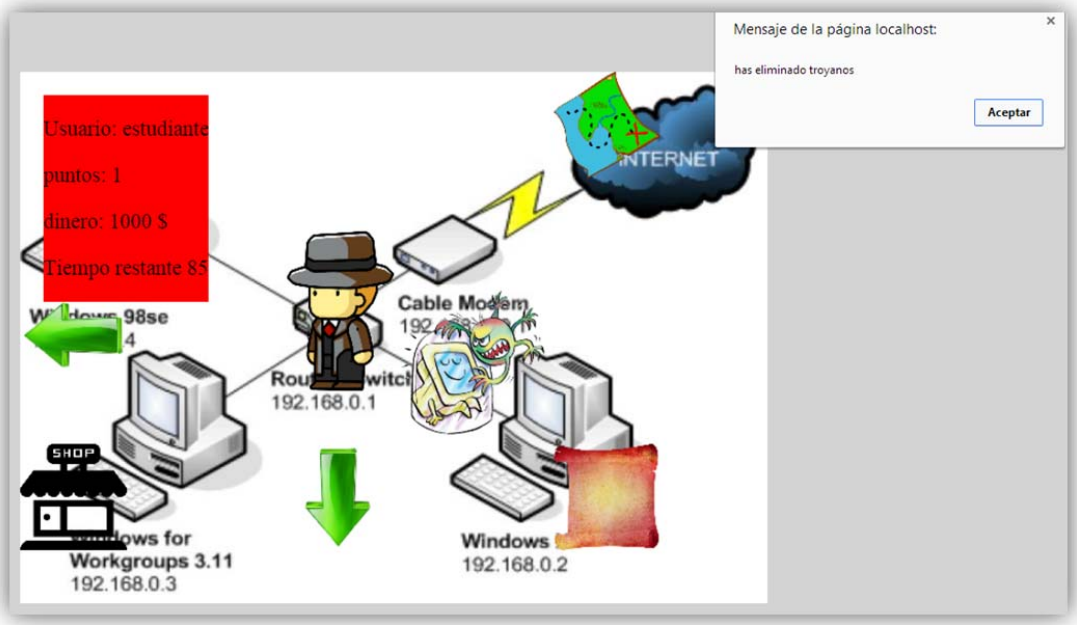



**Figura 78: Hacker Boy (7)**

En caso de no existir dichas amenazas, por no haber mirado el log donde debía informarse debidamente, el usuario pierde puntos, a lo que se avisa con el correspondiente mensaje indicativo.

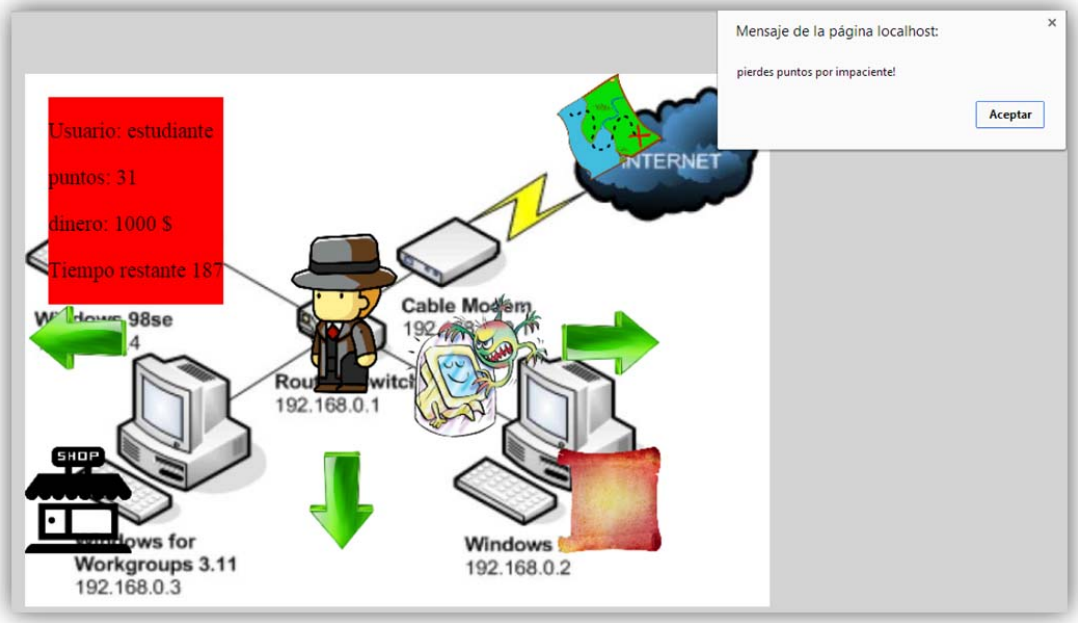



**Figura 79: Hacker Boy (8)**

Finalmente cuando se acaba el tiempo, se recibe información sobre las insignias obtenidas durante la partida.

## Sección 7.5. Bad rabbits die hard!

***Definición*** Juego orientado a la enseñanza de la programación con paralelismo entre la aplicación y las propiedades de la programación. Por medio de distintas pantallas se tratarán de enseñar algunos conceptos básicos de la programación, por ejemplo:

- Variables y tipos
- Records
- Funciones con y sin parámetros
- Arrays, matrices
- Sentencias if, bucles

En cada una de las pantallas se debe conseguir evitar la muerte de los conejos mediante uso de herramientas inherentes a la pantalla que simulan elementos característicos de la programación. Se evita el uso de código dado que está orientado no a aprender un lenguaje de programación sino los conceptos básicos de la misma.

***Desarrollo:*** Inicialmente, el usuario elegirá el nivel que quiere jugar.

Cada pantalla/nivel tienen en común:

1) Somos un conejo con capacidad limitada de hacer magia.
2) Posibilidad de morir por parte del conejo.

El objetivo de cada pantalla es evitar la muerte del conejo. En todo momento hay un botón de abandono de manera que el conejo se suicida. El objetivo de este botón es el de dar punto de retorno en el momento en el que la magia se ha agotado por parte de los conejos.

Existen explicaciones en cada pantalla de manera que cuando un jugador no sepa un concepto pueda visualizarlas, o aprenda los tipos de conjuros que sabe el conejo al que intentaremos salvar, por lo que será necesario documentarse en cada nivel. Esta documentación no será en ningún caso extensa en ninguno de los niveles.

La posibilidad de hacer magia antes de que se agote simula un acto de programación, como si se estuviese implementando una función a la que el usuario fija los valores de los parámetros, de manera que se puedan obtener resultados tanto positivos como negativos para el conejo de la acción de hacer magia.

**Insignias**

**Tabla 14: Insignias Bad Rabbits die hard**

| ID | NOMBRE | DESCRIPCIÓN | PUNTOS |
|---|---|---|---|
| 301 | DOMINA VARIABLES | genera tus primeras variables | 100 |
| 302 | DOMINA FUNCIONES | genera tus primeras funciones | 100 |

| | | | |
|---|---|---|---|
| 303 | SUICIDA | perder a veces no es tan malo | 50 |
| 304 | IF | domina sentencias if | 100 |
| 305 | LOOPY LOOP | genera tus bucles | 150 |
| 306 | ARRAY | domina arrays | 100 |
| 307 | MAGICIAN | acaba un nivel | 50 |
| 308 | 1BY1 | acaba el modo historia | 450 |
| 309 | MENDEL | domina herencia | 200 |

***Ejemplo de partida:***

El usuario se encuentra inicialmente en una situación en la que cabe la posibilidad de que uno o varios conejos dispersos por la pantalla pueden morir. El objetivo es salvarlos. En todo momento tiene tres botones con diferentes opciones, instrucciones, rendirse o hacer magia.

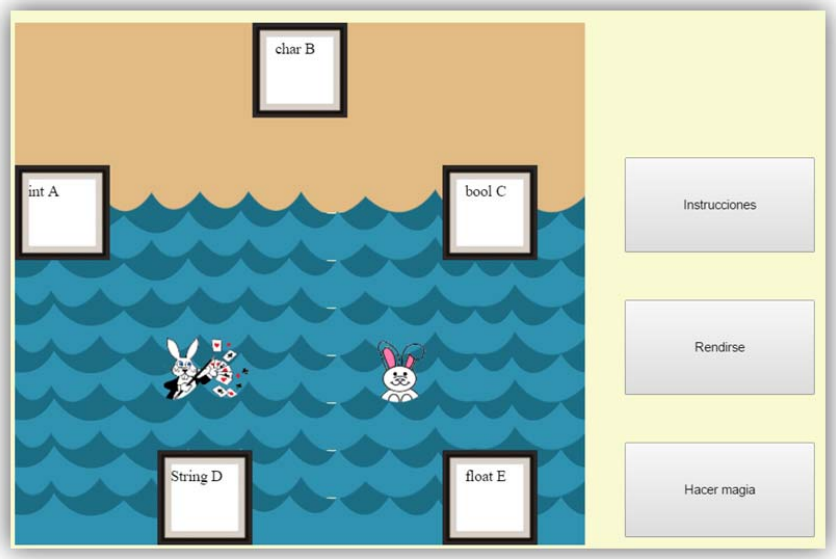



**Figura 80: Bad Rabbits die hard**

A menos que el usuario haya jugado al nivel en el que se encuentra, se recomienda que inicialmente se pulse el botón de Instrucciones para localizar objetivos del nivel.

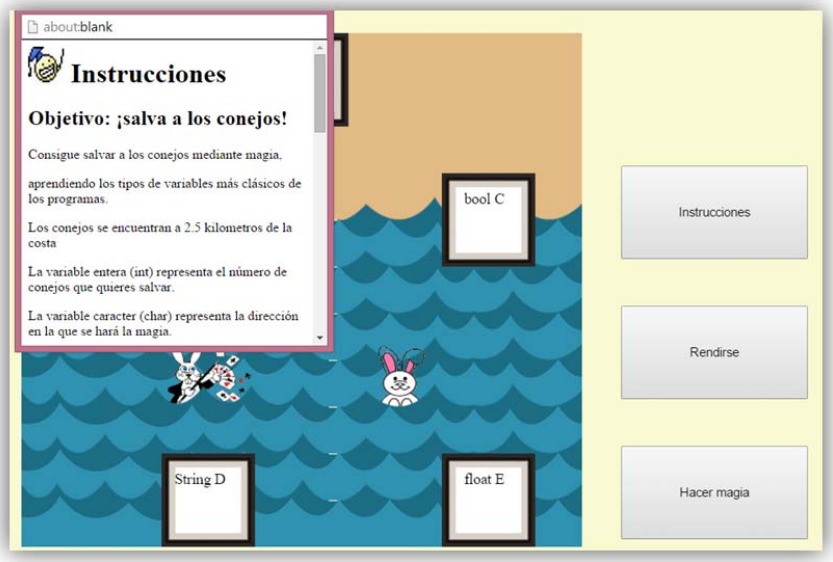



**Figura 81: Bad Rabbits die hard**

En el momento en que los usuarios se rinden, los conejos no sobreviven.


**Figura 82: Bad Rabbits die hard**

En el caso de usar la magia pueden darse tres diferentes situaciones. La primera de ellas, que la magia esté mal ejecutada. En dicho caso, los usuarios no sobreviven.


**Figura 83: Bad Rabbits die hard**

Otra de las situaciones posibles es que sea la última pantalla antes del final, y, que al estar bien ejecutada la magia, los conejos sobrevivan.


**Figura 84: Bad Rabbits die hard**

La última opción es que lleve a una pantalla intermedia, en dicho caso un mensaje indicativo con las características de la nueva pantalla aparecerá. (En la imagen se observa que aún no está cargada la imagen de la siguiente pantalla en el momento que aparece el mensaje, pero aparecerá gracias al bucle de repintado que se explicará en el capítulo 8)


**Figura 85: Bad Rabbits die hard**

## Sección 7.6. Circuits

***Disponibilidad:*** Aplicación de escritorio.

***Definición:***

El juego muestra un pequeño circuito de puertas lógicas, y las entradas generadas aleatoriamente (e.g. 0 1 1 0). El jugador debe decir en menos de una cantidad determinada de segundos (inicialmente esa cantidad será de 10 segundos y se irá incrementando en función del nivel de dificultad del circuito) si la salida es 0 ó 1.

***Desarrollo del juego:***

El juego se compone de 10 escenarios predefinidos (dibujos de circuitos lógicos). Se empieza por el primero y el jugador tendrá que resolver cada escenario mediante 5 intentos con valores de entrada aleatorios y por tanto diferentes en cada uno de ellos.

Después, otras 5 entradas para el segundo, etc.

- El jugador comenzará el juego con un total de tres vidas y puntuación inicial igual a cero.
- Por cada acierto que consiga, se sumarán <100*nivel> puntos.
- Por cada fallo se pierde una vida. Al tercer fallo, la partida finaliza.

***Insignias obtenibles***

Basadas en puntuación en una única partida (no se acumulan puntos entre partidas

Tabla 15: Insignias del juego Circuits

| ID | NOMBRE | DESCRIPCIÓN | PUNTOS |
|---|---|---|---|
| 401 | RONDA CON PLENO | Una insignia por conseguir hacer "pleno" en cualquiera de los circuitos (solamente se concede una vez, da igual en qué nivel sea) | 50 |
| 402 | 20 DE GOLPE | Conseguir 20 aciertos seguidos | 100 |
| 403 | PRINCIPIANTE | Alcanzar los 7.500 puntos en una partida | 250 |
| 404 | AVANZADO | Alcanzar los 18.000 puntos en una partida | 400 |
| 405 | EXPERTO | Alcanzar los 27.500 puntos en una partida | 500 |

***Ejemplo de partida:***

Al igual que en los juegos Java anteriores, Circuits posee una ventana para que el jugador inicie sesión en el juego (Figura 86), otra ventana en la que podrá acceder a nuestra web para registrarse si todavía no lo ha hecho y una ventana inicial presentando el comienzo de la partida (Figura 87).

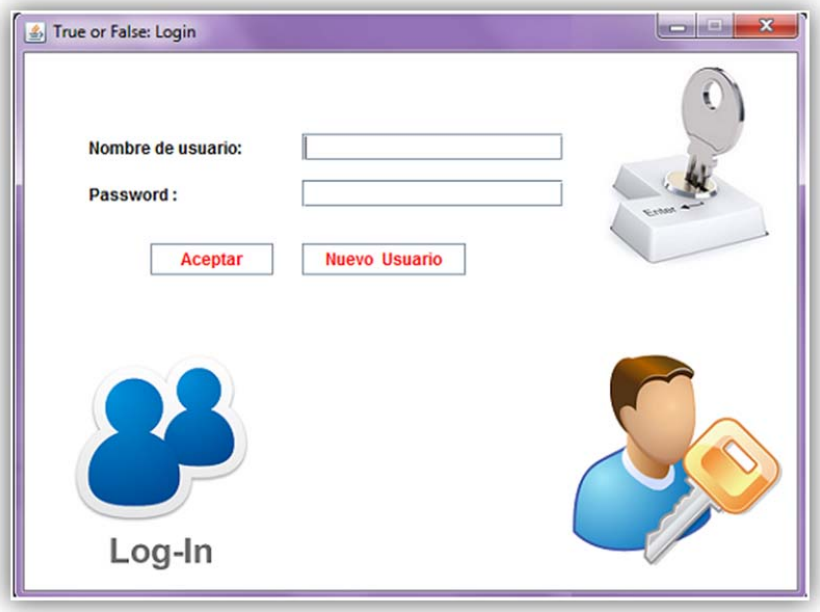



Figura 86: Ventana de Login del juego Circuits para aplicación de escritorio

A continuación, mostramos la ventana inicial que presenta el juego:


**Figura 87: Ventana inicial del juego Circuits para aplicación de escritorio**

El juego comenzará con circuitos lógicos sencillos e incrementará su dificultad poco a poco a medida que el usuario vaya superando los diferentes niveles (se añaden más y nuevos componentes en los circuitos cada vez que el usuario logra superar un escenario).

Tras confirmar que el jugador quiere comenzar su partida, aparecerá el primer nivel (Figura 88) con un valor aleatorio para las entradas y con un tiempo inicial igual a 10 segundos (dicho tiempo se irá decrementando). Antes de que ese tiempo finalice, el jugador deberá seleccionar una de las dos opciones, 0 ó 1, y seguidamente pulsar el botón de Resolver para ver si es resultado ha sido o no correcto.

- En la **parte izquierda** se sitúan las cuatro entradas de los circuitos. Estas cambian su valor de manera continua y aleatoria en cada intento perteneciente a un nivel.
- En la **parte central** se representará el circuito perteneciente al nivel de juego actual
- En la **parte derecha** aparecerá un marcador con el tiempo restante, las dos posibles salidas entre las que tenemos que elegir como resultado y un botón para confirmar y comprobar nuestra respuesta.

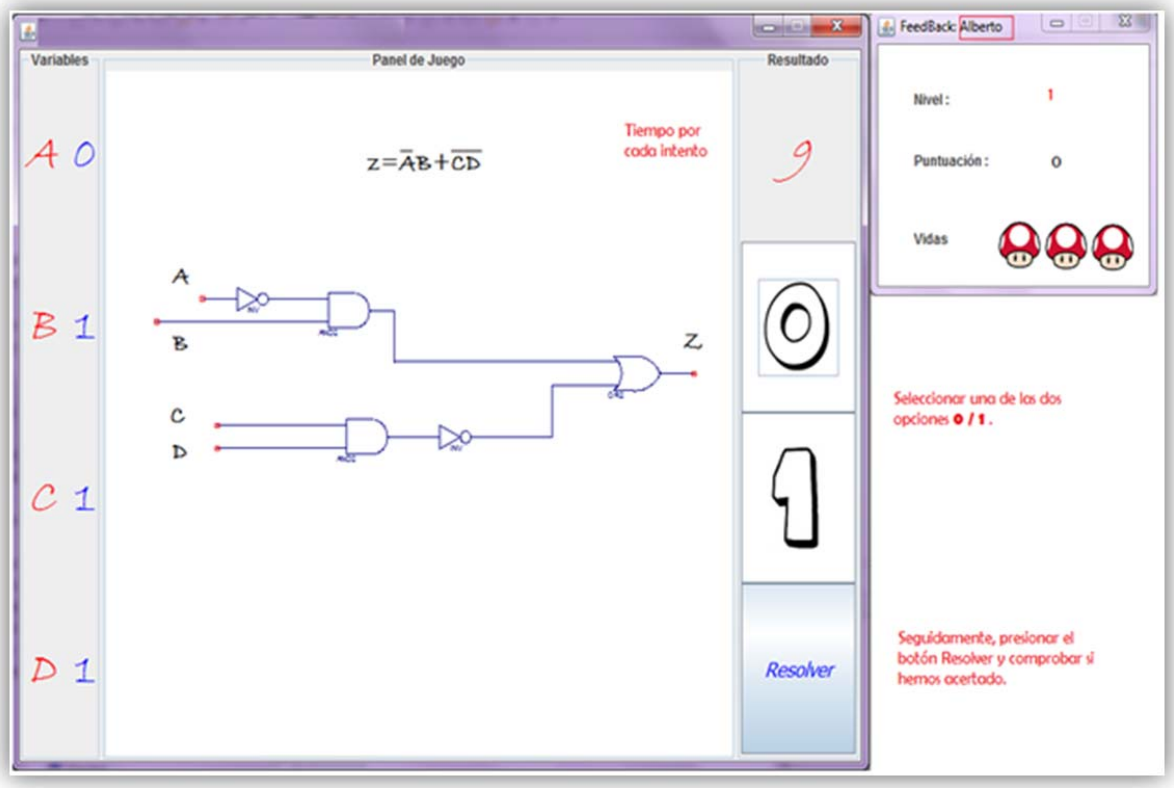



**Figura 88: Ventana principal del juego Circuits para aplicación de escritorio**

El jugador recibirá una notificación en la pantalla en la que podrá ver si su respuesta ha sido correcta o no. En consecuencia, sumará la cantidad correspondiente de puntos en su marcador o perderá una vida. El juego continuará hasta que éstas se agoten o responda de manera correcta a todos los circuitos (Figura 89).

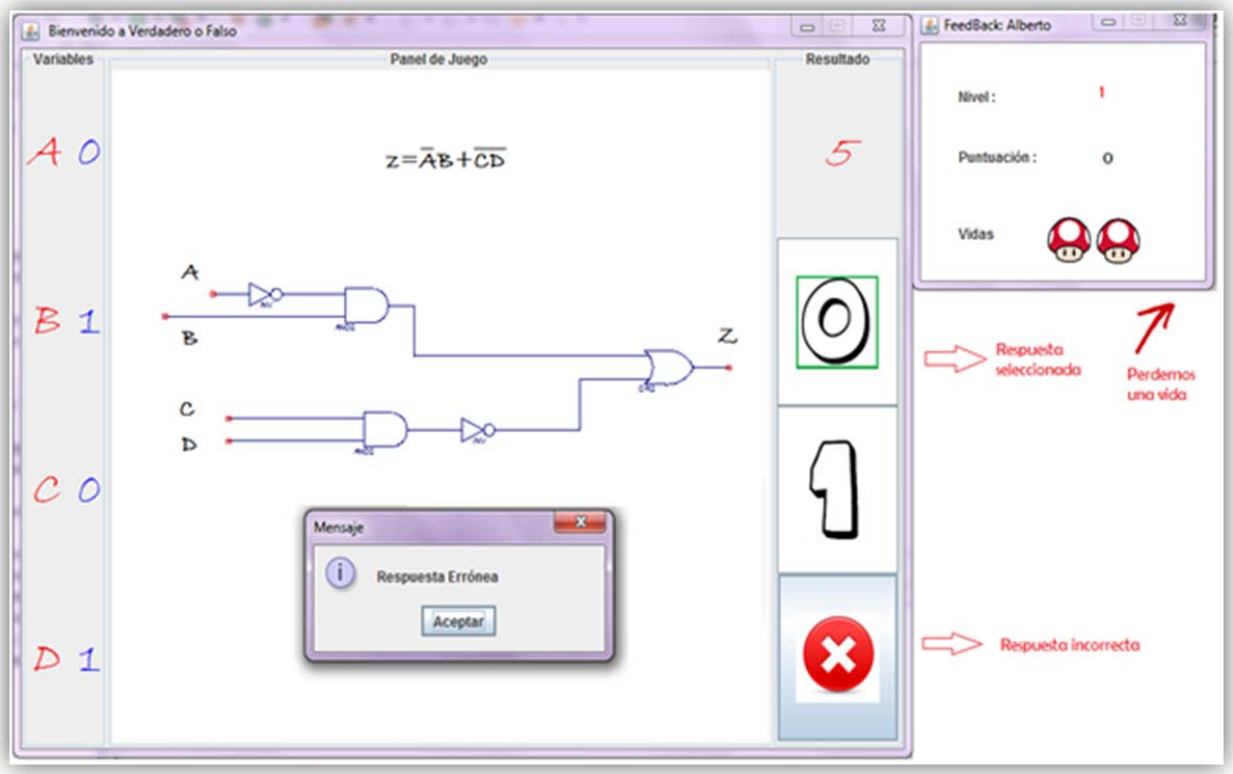



**Figura 89: Ventana del juego Circuits para tras pulsar el botón Resolver**

Por último, nos aparecerá una ventana en la que podremos ver que logros hemos desbloqueado en esta partida y el jugador tendrá la opción de abandonar o de volver a jugar si así lo desea.

# Bloque III
# Detalles de implementación

# Capítulo 8. Implementación: Infraestructura de servicios backend

En este capítulo trataremos la implementación del servidor y todos lo relacionado con el mismo.

Comenzaremos hablando del servidor y la configuración realizada para el desarrollo del proyecto. A continuación pasaremos a hablar de las tecnologías utilizadas, y los motivos de las decisiones tomadas. Finalmente hablaremos de la base de datos utilizada para la persistencia del servidor, terminando por dar una breve explicación de la estructura utilizada con las imágenes de la base de datos y las tablas creadas.

La arquitectura utilizada es cliente-servidor. En esta arquitectura podemos diferenciar por un lado los clientes, que serán los consumidores de los servicios y por otro lado el servidor que será el expositor de los servicios.

Estructuralmente, este capítulo consta de varios apartados. En primer lugar hablaremos del servidor que vamos a utilizar, lo que necesitaremos para ponerlo en marcha y las decisiones de implementación tomadas con explicaciones de cada una. A continuación pasaremos a hablar de la base de datos, comenzaremos explicando la base de datos utilizada y la herramienta usada para manejarla. A continuación explicaremos la implementación de la base de datos desarrollada, mostrando las tablas y la creación de las mismas. Finalmente hablaremos de los servicios expuestos en la API.

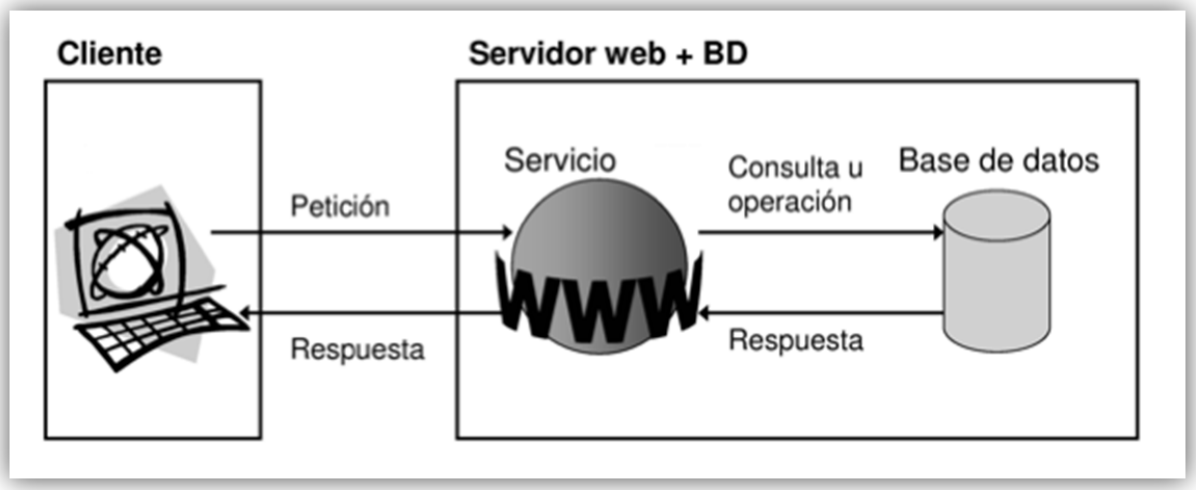



**Figura 90: Estructura del servidor**

## Sección 8.1. Servidor web

El servidor que utilizaremos pertenece a la Universidad Complutense. La propia facultad nos facilitará un sistema operativo Linux, Ubuntu Server, el cual nos será bastante útil ya que a pesar de no tener interfaz gráfica tiene un manejo no demasiado complicado.

Sobre este sistema operativo, entregado limpio, instalaremos todo lo necesario para poder poner en marcha los servicios que necesitamos. Los servicios web que decidimos implementar, tras estudiar que se adapta más a nuestras necesidades, fueron servicios REST (Representational State Transfer o Transferencia de Estado

Representacional). Con los servicios REST desarrollados en Java conseguimos una buena interfaz para XML y HTTP manteniendo cierta simplicidad que nos permite entender todo correctamente. Estos servicios cumplen varias reglas de diseño que son claves:

- Un protocolo cliente/servidor sin estado: cada mensaje HTTP contiene toda la información necesaria para comprender la petición. Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes.
- Un conjunto de operaciones bien definidas que se aplican a todos los *recursos* de información. Siendo en este caso JSON del que hablaremos más adelante.
- Una sintaxis universal para identificar los recursos. En un sistema REST, cada recurso es direccionable únicamente a través de su URI.

Todo esto nos permite que los servicios sean comunes para las distintas plataformas implementadas, que se pueda llamar desde java, html o android, únicamente invocando la URI expuesta. Esto nos será muy útil para poder implementar juegos similares en distintas plataformas ya que gran parte de la lógica sin depender de la plataforma la realiza el servidor.

Las operaciones se han realizarán con JSON (JavaScript Object Notation), un formato que aporta ligereza para el intercambio de datos y que a su vez no requiere el uso de XML. La motivación de utilizar JSON principalmente es la simplicidad que tiene su parseo para obtener los datos que contiene, teniendo en algunas plataformas métodos ya diseñados para ello.

Estructura básica del JSON:

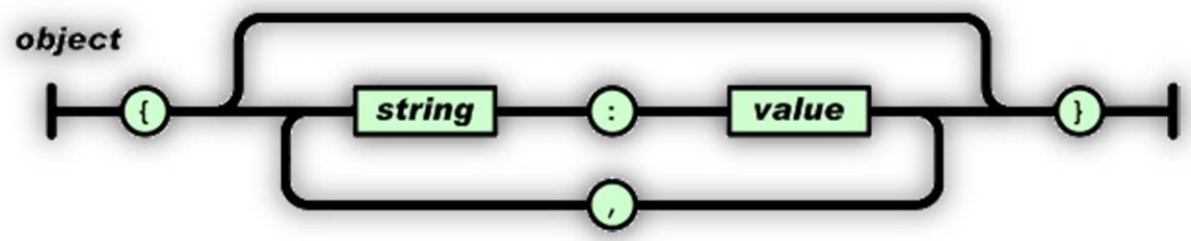



**Figura 91: Estructura básica de JSON (Fuente: http://json.org)**

Una vez definido el tipo de servicios que queríamos utilizar pensamos en varios servidores de aplicaciones que nos podían ayudar para desplegarlos. Por los conocimientos que teníamos descartamos algunos y nos quedamos con Apache Tomcat y JBoss Application Server. Como varios miembros del grupo ya habían trabajado con Jboss decidimos finalmente utilizar el JBoss Application Server ya que conocíamos más funcionalidades y además estábamos más familiarizados con el entorno gráfico de configuración que ofrece. Con esto ya teníamos el comienzo para poder desplegar nuestros servicios. Como éramos conscientes que toda la lógica que realizábamos en los servicios necesitaría una capa de persistencia, decidimos introducir una base de datos que diera servicio a nuestro desarrollo.

## Sección 8.2. Base de datos

La base de datos introducida fue PostgresSQL, base de datos orientada a objetos, relacional y libre, publicada bajo la licencia BSD. Para poder acceder a ella desde el código Java utilizamos una herramienta de mapeo

Objeto-Relacional, Hibernate. Esta herramienta, de código libre también bajo una licencia GNU LGPL, es flexible en cuanto al esquema de tablas utilizado, lo cual nos ayudaba bastante ya que la definición de nuestra base de datos sufrió varios cambios a lo largo del proyecto. La manera en la que utilizamos Hibernate fue creando beans para cada tabla a la que necesitábamos acceder y añadiendoles las anotaciones correspondientes.

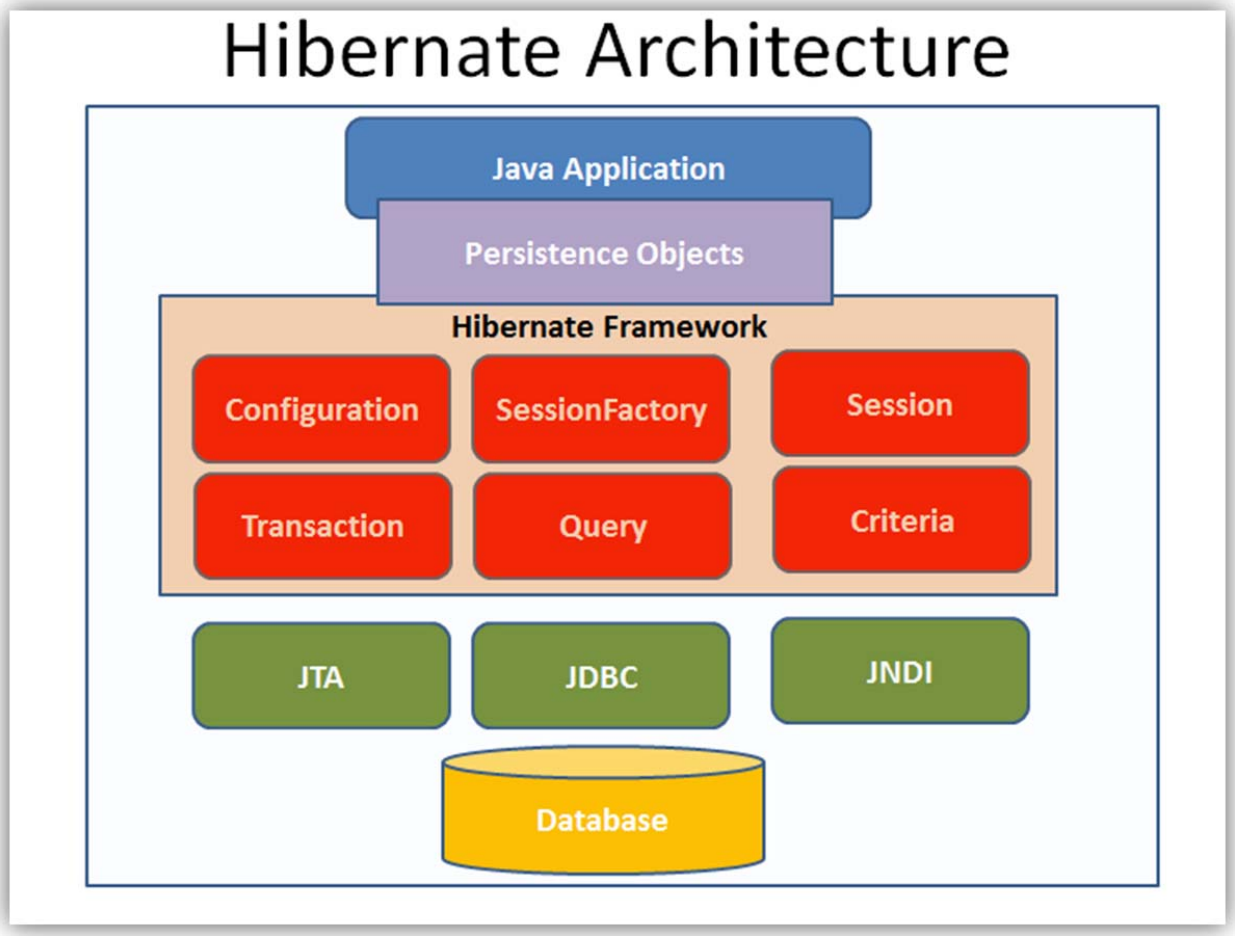



**Figura 92: Arquitectura de Hibernate (Fuente: http://www.onlinetechvision.com/hibernate-overview/)**

Para la gestión/acceso desde sistema operativo hemos utilizado dos maneras distintas:

- Desde Ubuntu (propio servidor), acceso a través de la consola ofrecida por PostgreSQL, psql, desde la cual podíamos realizar cualquier acción a través de comandos
- Desde Windows (una máquina externa), PgAdmin, un entorno gráfico que nos permite realizar cualquier acción de una manera más amigables y teniendo una visión más amplia de la base de datos.

Finalmente configuramos el servidor para que fuera accesible desde otros equipos, ya fuera desde otro equipo Linux con un comando ssh o desde un equipo Windows a través del último software del que hablaremos en este apartado, el Putty.

Putty es un cliente ssh para Windows libre. Con el conseguimos acceder al servidor a nivel consola para realizar cualquier acción y a la vez nos permitía realizar tunneling para poder acceder al puerto de la base de datos con el PgAdmin. Este software nos facilitó mucho la gestión de la base de datos, ya que se explicaba anteriormente, podíamos tener un entorno gráfico que nos daba una visión más amplia de la misma.

### Sección 8.2.1. Implementación de la base de datos

Para las necesidades de toda la infraestructura hemos creado una base de datos en la cual persistimos todos los datos necesarios.

La base de datos está dirigida a guardar los usuarios, los grupos a los que pertenece cada usuario y las insignias logradas. Esta base de datos contiene todo necesario para la infraestructura, la web y la parte más social. A continuación mostraremos la base de datos, directamente relacionada con el capítulo en el que estamos. De la base de datos de juegos hablaremos en el capítulo relacionado con los juegos (Capítulo 8).

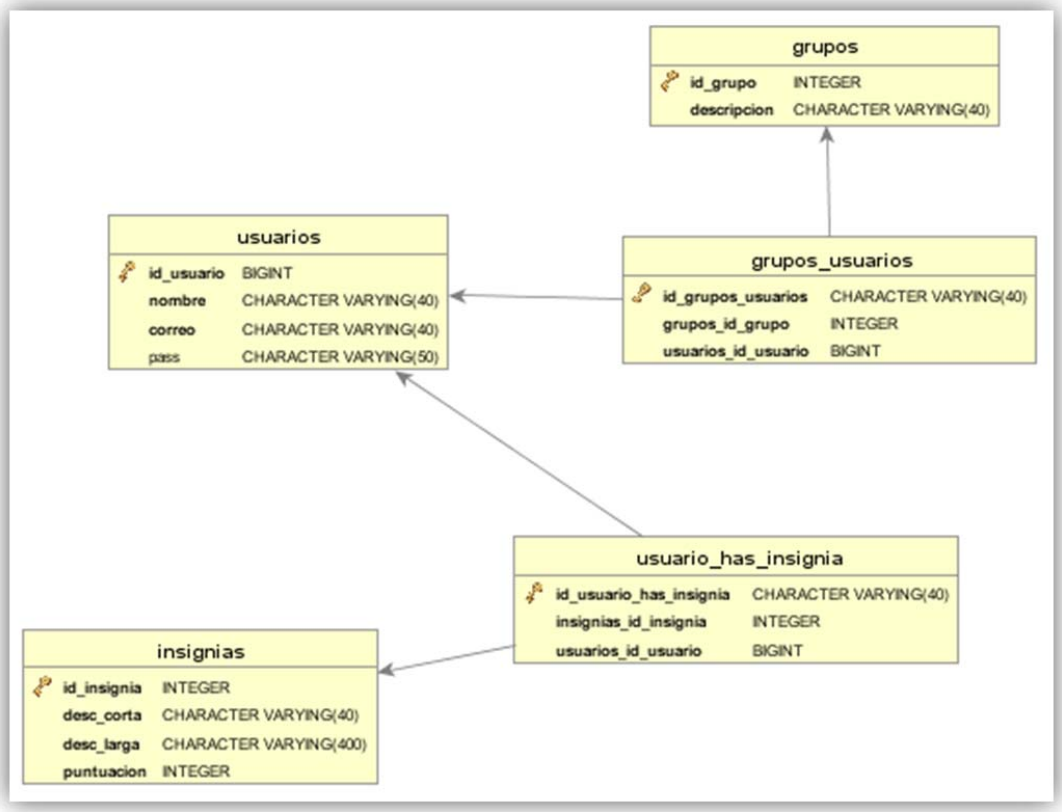



**Figura 93: Esquema de base de datos InsigniasFDI**

## *Tabla Usuarios*

Define los usuarios que existen, tiene un identificador único para cada usuario que será utilizado para relacionarlo con el grupo al que perteneces y con las insignias que consiga. También contiene datos básicos del usuario, como el nombre, el correo y la contraseña de acceso.

**Tabla 16: Ejemplo usuarios en base de datos**

| id_usuario bigint | nombre character varying(40) | correo character varying(40) | pass character varying(50) |
|---|---|---|---|
| 1419951352244 | User1 | user1@ucm.es | pass1 |
| 1421011435564 | Miguel | miguel@olimpiada.fdi.ucm.es | miguel |
| 1421059164777 | p12 | p12@ucm.es | aaaa |
| 1421577777882 | fails | fails@javi.yo | javi |

```
CREATE TABLE usuarios (
    id_usuario bigint NOT NULL,
    nombre character varying(40) NOT NULL,
    correo character varying(40) NOT NULL,
    pass character varying(50)
    CONSTRAINT usuarios_pkey PRIMARY KEY (id_usuario)
)
```

**Figura 94: Código de creación de tabla usuarios**

## *Tabla Grupos*

Define los grupos que existen, tiene un identificador único para cada grupo que será utilizado para relacionarlo con los usuarios que pertenezcan al grupo. También contiene un nombre de grupo.

**Tabla 17: Ejemplo de grupos en base de datos**

| id_grupo integer | descripcion character varying(40) |
|---|---|
| 1 | Grupo one |
| 2 | Grupo two |

```
CREATE TABLE fdi.grupos
(
  id_grupo integer NOT NULL,
  descripcion character varying(40) NOT NULL,
  CONSTRAINT grupos_pkey PRIMARY KEY (id_grupo)
)
```

**Figura 95: Código de creación de tabla grupos**

## *Tabla Insignias*

Define las insignias que existen, tiene un identificador único para cada insignia que será utilizado para relacionarlo con los usuarios que consigan la insignia. También contiene una descripción corta, que mostrará una breve información, una descripción larga y la puntuación que obtendrá el usuario por la insignia.

**Tabla 18: Ejemplos insignias en base de datos**

| id_insignia integer | desc_corta character varying(40) | desc_larga character varying(400) | puntuacion integer |
|---|---|---|---|
| 1 | REY | Todas las preguntas acertadas! | 50 |
| 2 | EL 7 MAGICO | Acertadas 7 preguntas | 20 |
| 3 | RAPIDO Y EFICAZ | 5 aciertos en menos de 30 segundos | 15 |
| 4 | GENIO EN PROGRAMACION | 3 preguntas acertadas de Programacion | 10 |
| 5 | GENIO EN BASES DE DATOS | 3 preguntas acertadas de Bases de datos | 10 |

```
CREATE TABLE fdi.insignias
(
  id_insignia integer NOT NULL,
  desc_corta character varying(40) NOT NULL,
  desc_larga character varying(400) NOT NULL,
  puntuacion integer NOT NULL,
  CONSTRAINT insignias_pkey PRIMARY KEY (id_insignia)
)
```

**Figura 96: Código de creación de la tabla insignias**

## *Tabla Usuario_has_Insignia*

Relaciona un usuario con una insignia, de esta manera conocemos que insignias han sido conseguidas por cada usuario. Tiene un identificador único para cada registro, y en el registro se guarda qué número de insignia se ha conseguido (único como explicamos en la tabla anterior) y que usuario la ha conseguido con su id único también.

**Tabla 19: Ejeplos usuario_has_insignia en base de datos**

| id_usuario_has_insignia character varying(40) | insignias_id_insignia integer | usuarios_id_usuario bigint |
|---|---|---|
| 1421577905230 | 106 | 1421577777882 |
| 1421577905269 | 104 | 1421577777882 |
| 1421577905299 | 102 | 1421577777882 |
| 1421577905326 | 103 | 1421577777882 |
| 1421577905357 | 101 | 1421577777882 |
| 1421577905394 | 105 | 1421577777882 |
| 1421578182348 | 106 | 1421577777882 |

```
CREATE TABLE fdi.usuario_has_insignia
(
  id_usuario_has_insignia character varying(40) NOT NULL,
  insignias_id_insignia integer NOT NULL,
  usuarios_id_usuario bigint NOT NULL,
  CONSTRAINT usuario_has_insignia_pkey PRIMARY KEY (id_usuario_has_insignia),
  CONSTRAINT insignias_id_insignia_fkey FOREIGN KEY (insignias_id_insignia)
      REFERENCES fdi.insignias (id_insignia) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT usuarios_id_usuario_fkey FOREIGN KEY (usuarios_id_usuario)
      REFERENCES fdi.usuarios (id_usuario) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
)
```

Figura 97: Código de creación de la tabla usuario_has_insignia

### *Tabla Grupos_Usuarios*

Relaciona un usuario con un grupo, de esta manera conocemos que usuarios pertenecen a un grupo. Tiene un identificador único para cada registro, y en el registro se guarda qué número de grupo y que usuario pertenece a ese grupo.

Tabla 20: Ejemplo de grupos_usuarios en base de datos

| id_grupos_usuarios character varying(40) | grupos_id_grupo integer | usuarios_id_usuario bigint |
|---|---|---|
| 1419953037027 | 2 | 1419951352244 |

```
CREATE TABLE fdi.grupos_usuarios
(
  id_grupos_usuarios character varying(40) NOT NULL,
  grupos_id_grupo integer NOT NULL,
  usuarios_id_usuario bigint NOT NULL,
  CONSTRAINT grupos_usuarios_pkey PRIMARY KEY (id_grupos_usuarios),
  CONSTRAINT grupos_id_grupo_fkey FOREIGN KEY (grupos_id_grupo)
      REFERENCES fdi.grupos (id_grupo) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT usuarios_id_usuario FOREIGN KEY (usuarios_id_usuario)
       REFERENCESfdi.usuarios (id_usuario) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
)
```

Figura 98: Código de creación de la tabla grupos_usuarios

## Sección 8.3. Resumen de la API

En este capítulo se ha explicado todo lo relacionado con el servidor tratando de aportar los detalles necesarios para comprender la arquitectura y los motivos por los que se ha elegido esta arquitectura. Se ha

entrado en detalle en las partes que consideramos más importantes de nuestra arquitectura y que ayudan a que el desarrollo sea más sencillo y eficaz. A su vez se ha explicado los motivos por los que hemos tomado las decisiones a lo largo del desarrollo. A continuación exponemos todos los servicios que se han implementado para el funcionamiento de la infraestructura, estos servicios se encuentran detallados en el ANEXO A.

**Tabla 21: Servicios expuestos en la API del servidor**

| Servicio | Descripción |
|---|---|
| Insertar Usuario | Recibe los campos nombre, correo y contraseña y crea un usuario en base de datos |
| Insertar insignia | Recibe los campos id insignia, descripción corta, descripción larga y puntuación y crea una insignia en base de datos |
| Insertar grupo | Recibe los campos id grupo y descripción y crea un grupo en base de datos |
| Asignar insignia a usuario | Recibe el nombre de usuario y el id de la insignia que ha conseguido y lo persiste en base de datos, con esto asignamos la insignia al usuario |
| Asignar usuario a grupo | Recibe el nombre de usuario y el id del grupo al que debe añadirse el usuario y lo persiste en base de datos, con esto asignamos el usuario al grupo |
| Borrar usuario de grupo | Recibe el nombre de usuario y el id del grupo del que debe borrarse al usuario y lo elimina en base de datos, con esto eliminamos el usuario del grupo |
| Get insignias usuario | Recibe el nombre del usuario y accede a base de datos para devolver las insignias que el usuario recibido tiene conseguidas |
| Login usuario | Recibe el nombre de un usuario y la contraseña y comprueba si es correcto en base de datos para permitir al usuario hacer login o pedirle que repita las credenciales |
| Get puntos insignias grupo | Recibe el id de un grupo y devuelve la puntuación que tiene ese grupo por todas las insignias conseguidas por sus usuarios |
| Get puntos insignias usuario | Recibe el nombre de un usuario y devuelve la puntuación que tiene ese usuario por sus insignias conseguidas |
| Get grupo por usuario | Recibe el nombre de un usuario y devuelve el id del grupo al que pertenece |

| | |
|---|---|
| Get usuarios de grupo | Recibe el id de un grupo y devuelve los nombres de los usuarios pertenecientes a ese grupo |
| Obtener todos los grupos | No recibe nada por parámetro y devuelve una lista con todos los grupos existentes |
| Obtener todos los usuarios | No recibe nada por parámetro y devuelve una lista de todos los usuarios existentes |

# Capítulo 9. Implementación de la página web

En este capítulo se detalla la implementación del portal descrito en el capítulo 5.

Como front-end que comunica con los servicios que ofrece el servidor descrito, se desarrolla un portal de la Olimpiada FDI, interfaz con la que el usuario puede interactuar para acceder a diferentes características de la plataforma a modo de prueba de concepto.

La página tiene en cuenta no solo el inicio de sesión del usuario sino la plataforma desde la que está realizando dicha sesión, ya que de esta plataforma depende la interacción con la página y los servicios que ofrece.

Estructuralmente, este capítulo consta de varios apartados. En primer lugar, se explicarán las tecnologías que se han empleado con una explicación simple del porqué de su utilización. Seguidamente, se comentará la arquitectura de la web en detalle, incluyendo la manera en que el portal interacciona con el back-end que disponemos (servidor) y la manera en que se establece relación con los juegos de las diferentes plataformas. El siguiente apartado define la interfaz y sus diferentes plataformas, explicando cómo Bootstrap ha ayudado en la implementación. Para finalizar, el último apartado resumirá el uso del portal.

## Sección 9.1. Tecnologías empleadas

En lo referente a tecnología web, la tendencia de la tecnología sigue una evolución hacia el diseño simple y el diseño responsive, que evita el diseño específico para el móvil para centrarse en diseños adaptables.

HTML, como se ha definido en el capítulo 3 para su versión 5, es uno de los estándares de la web más conocidos y utilizados. Así, el uso de HTML está complementado por la utilización de CSS para estilo y JavaScript para funcionalidad interna.

Bootstrap es una herramienta muy útil a la hora del diseño simple, dado que ayuda a la implementación de una web con estructuras predefinidas. Por supuesto, en el apartado del diseño responsive tiene un rol fundamental ya que es la herramienta en la que está basada toda la estructura de la web, permitiendo adaptar los diferentes bloques de contenido en función del tamaño de la pantalla utilizada.

La estructura de la implementación de la página web ha sido realizada en función de la funcionalidad concreta de cada apartado de la misma. Se han usado como diferentes lenguajes de código php, JavaScript y HTML, y como herramienta de apoyo Bootstrap y hojas de estilo CSS, de manera que todos cumplían una función:

| | |
|---|---|
| HTML | Para funcionalidad básica del sistema, así como para la estructuración de las diferentes páginas del portal. La gestión de las imágenes o textos se ha implementado mediante HTML |

| | |
|---|---|
| CSS | La utilización de las páginas de estilo en el caso de la página web ha sido mínima, debido a que se han soportado estilos principalmente por medio de la herramienta Bootstrap, sin embargo, para colocación de ciertos bloques de contenido se ha aprovechado su utilidad. Comentar que este apartado referencia a CSS definidos por nosotros y no a los incluidos en Bootstrap por defecto. |
| PHP | Para funcionalidad interna a la página, como almacenamiento de sus diferentes variables, siempre en contacto con JavaScript |
| JavaScript | Para la conexión con el back-end. Toda conexión con los servicios ha utilizado scripts sobre los que se ha realizado la misma. |
| Bootstrap | Utilizado para colocación de bloques y para interfaces en diferentes plataformas. Ha permitido el diseño responsive de la página. |

## Sección 9.2. Arquitectura de la web

Las funcionalidades de la web están clasificadas en dos bloques: el bloque de juegos y el bloque de servicios. Recordando la figura que describe el funcionamiento completo de la plataforma (ver figura inferior) el portal es uno de los dos posibles puntos de entrada del sistema (siendo el otro los propios juegos):

En lo referente al bloque de juegos, dependiente del juego en cuestión, se puede descargar el juego para su modo offline. En dicho caso, se deberá loguear correctamente para el juego, como se explicará en el capítulo 8. Para algunos juegos, se puede jugar online (en caso de tener una implementación web). Para estos no es necesario hacer log in, dado que ya se ha realizado al entrar a la página.

En lo referente a la implementación, tal y como muestra la figura, la web lanza los juegos que estén en formato web e indexa otros para su descarga, por lo que el volumen del portal es amplio.

El bloque de servicios es aquel para el que el usuario sí necesita del portal. Así, para poder observar clasificaciones o participar con otros usuarios como miembros de un grupo, es necesaria la interacción con la página. Ésto se ha explicado en detalle en el capítulo 5.

En lo referente a la implementación, para el acceso a los servicios se realizan conexiones mediante php con el backend, almacenando los resultados en variables, todo dentro de scripts controlados que se ejecutan en función del servicio que queramos pedir. Se almacenan los datos obtenidos para ser procesados por JavaScript, manteniendo el propósito general de cada tecnología.

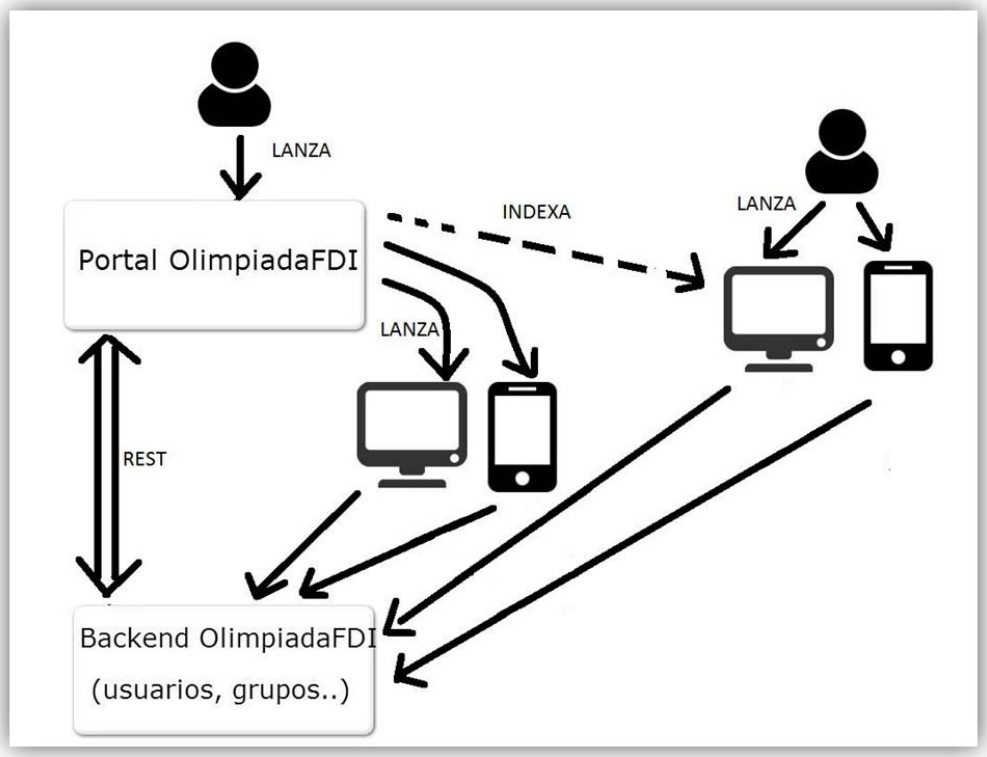



**Figura 99: Arquitectura de la web**

## Sección 9.3. Interfaz para las diferentes plataformas

Ayudados por la herramienta Bootstrap, hemos implementado un sistema de diferenciación de interfaces acorde al navegador en el que se pueda acceder al portal.

La diferenciación permite distinguir entre escritorio, y plataforma móvil. Está basada en los tamaños de las pantallas de cada navegador concreto.

En el caso de acceder a la plataforma desde un móvil, se hará incómoda la aparición de dos secciones de la página. Recordando cómo está implementado sobre escritorio:

Para el caso de la plataforma móvil se hace incómoda la iteración. Debido a la implementación de Bootstrap, gracias a su diseño responsive, se redimensionan los contenedores y, por lo tanto, se reordenan si el tamaño de pantalla disminuye lo suficiente.


**Figura 100: Distribución de la web vista desde un PC**


**Figura 101: Distribución de la web visita desde un dispositivo móvil**

# Capítulo 10. Implementación: Juegos

Después de describir la implementación de la página web, interfaz desde la que accedemos a la funcionalidad interna de la plataforma, en este capítulo describiremos la funcionalidad más visible del sistema: los juegos implementados. Éstos, se han desarrollado en diferentes plataformas, y son un modelo de la potencia que pudiera llegar a tener el sistema. Están en todo caso orientados a la enseñanza de la ingeniería informática, tal y como se refleja en el capítulo de introducción.

Este capítulo está estructurado de manera que inicialmente se identificarán las características comunes a todos los juegos. Para continuar, incluimos un apartado dedicado a los servicios REST expuestos en el backend de juegos. A esta parte sigue una declaración de la implementación de los juegos desarrollados en todos los entorno. Para concluir, se escribirá un resumen final del capítulo incluyendo las partes más importantes de los juegos.

## Sección 10.1. Características

Cada juego implementado se ha creado para servir como modelo de juego real, de manera que no existen versiones de prueba de los juegos, están todos completados en cuanto a implementación.

La característica principal que cumplen todos los juegos del entorno es la orientación a la enseñanza de la informática, de manera que los usuarios aprendan jugando y, a su vez, los usuarios más experimentados puedan obtener insignias de manera más sencilla a usuarios inexpertos debido a sus conocimientos.

- Una característica muy reseñable es que, siguiendo el modelo de las plataformas de servicios multijugador, se incluyen insignias de manera obligatoria en todos los juegos. En el siguiente apartado se declararán en profundidad las insignias.

## Sección 10.2. Bases de datos para juegos

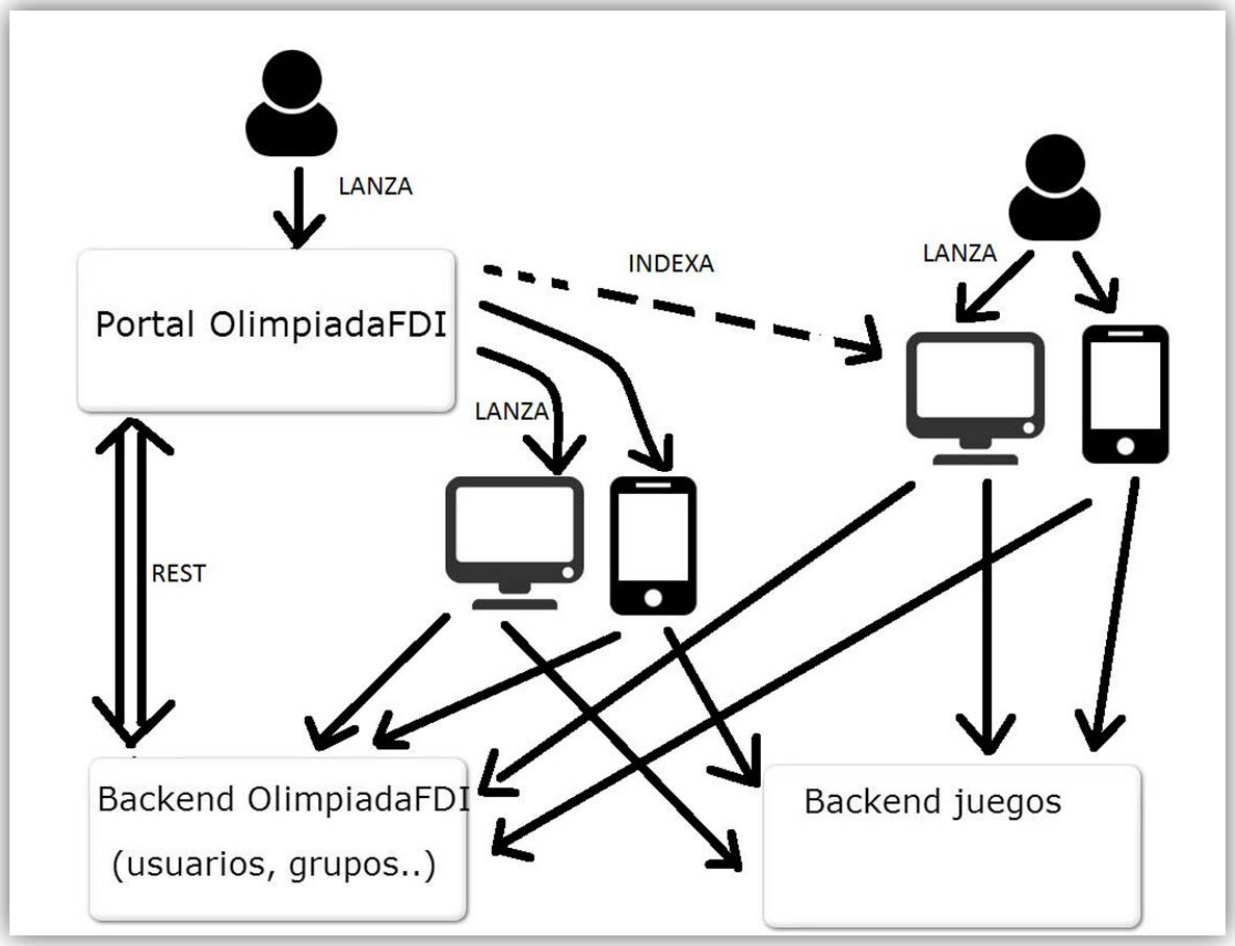



**Figura 102: Arquitectura de la plataforma**

Además del backend de servicios existe un backend orientado en exclusiva al uso de los juegos. Se definirán los servicios ofrecidos por el mismo en la sección a continuación.

### Sección 10.2.1. Base de datos BDOlimpiadaFDI (Juegos)

Para las necesidades de los juegos hemos creado una base de datos que se encarga de manejar todos los datos que son necesarios para el correcto funcionamiento de los juegos.

La base de datos está dirigida a guardar las preguntas, respuestas y toda la información necesaria para los juegos implementados. Esta base de datos está creada en PostgreSQL y para el manejo desde la API de juegos utilizamos Hibernate. A continuación mostraremos la base de datos.

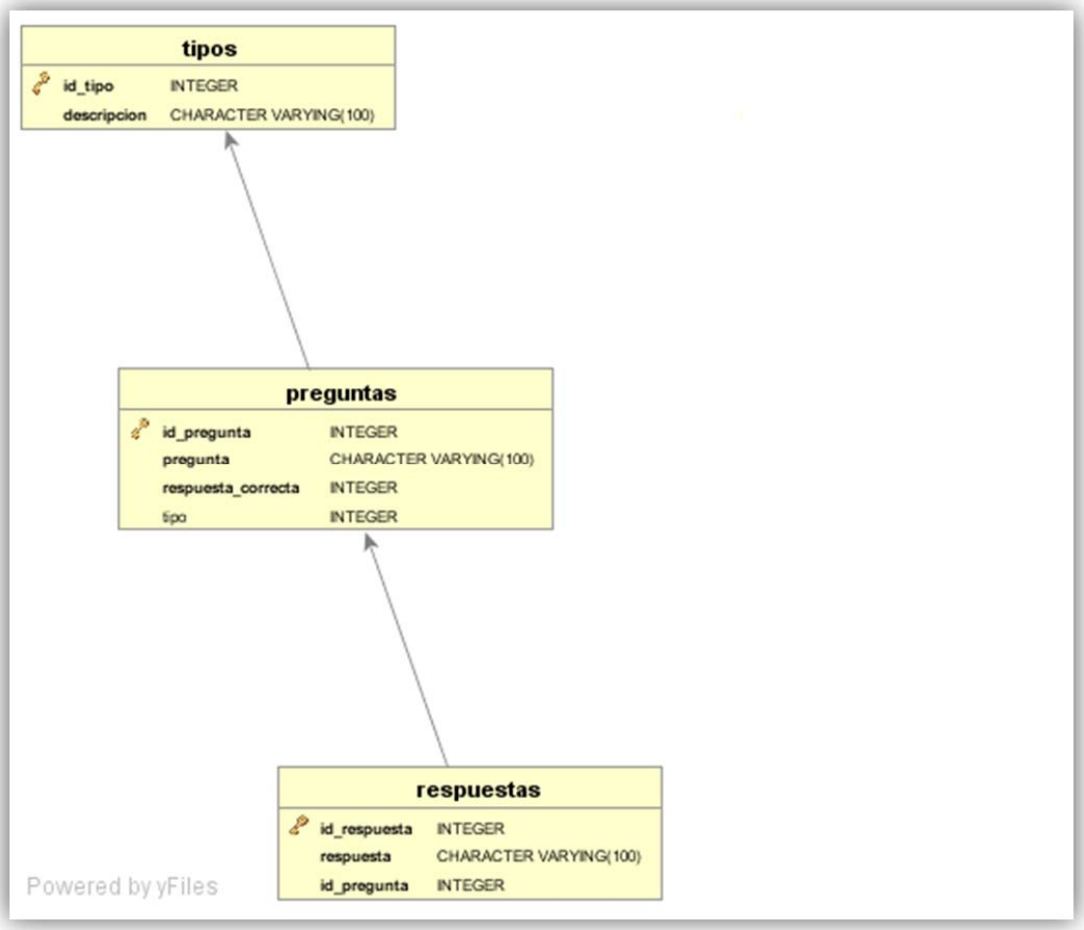



**Figura 103: Esquema de base de datos OlimpiadaFDI**

## *Tabla Tipos*

Define los tipos de preguntas que existen, tiene un identificador único para cada tipo y una descripción.

**Tabla 22: Ejemplo de tipos en base de datos**

| id_tipo integer | descripcion character varying(100) |
|---|---|
| 1 | Programacion |
| 2 | I.S. |
| 3 | Bases |
| 4 | Redes |
| 5 | S.O. |
| 6 | Computadores |

Figura 104: Código de creación de tabla tipos

```
CREATE TABLE fdi.tipos
(
  id_tipo integer NOT NULL,
  descripcion character varying(100) NOT NULL,
  CONSTRAINT tipos_pkey PRIMARY KEY (id_tipo)
)
```

## *Tabla Preguntas*

Contiene las preguntas que se usan en los juegos. Tiene un identificador único para cada pregunta, un campo donde se introduce la pregunta, a continuación otro campo donde se pone el identificador único de la respuesta que hace referencia a la tabla explicada a continuación y finalmente un campo tipo que hace referencia a la tabla explicada anteriormente

Tabla 23: Ejemplos de preguntas en bases de datos

| id_pregunta integer | pregunta character varying(100) | respuesta_correcta integer | tipo integer |
|---|---|---|---|
| 1 | DNS... | 2 | 4 |
| 2 | Firewall | 7 | 4 |
| 3 | Definicion de footprinting | 12 | 4 |
| 4 | El phising | 16 | 4 |
| 5 | ¿Cual es el estandar de facto de escaneo de los puertos? | 20 | 4 |
| 6 | ¿Cual es el protocolo mas efectivo a la hora de enviar video en directo? | 21 | 4 |
| 7 | Estados posibles, definiendo reglas en ipTable,s segun el estado de la conexion | 27 | 4 |

```
CREATE TABLE fdi.preguntas
(
  id_pregunta integer NOT NULL,
  pregunta character varying(100) NOT NULL,
  respuesta_correcta integer NOT NULL,
  tipo integer, CREATE TABLE fdi.preguntas
(
  CONSTRAINT preguntas_pkey PRIMARY KEY (id_pregunta),
       CONSTRAINT tipo_fkey FOREIGN KEY (tipo)
      REFERENCES fdi.tipos (id_tipo) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
)
```

Figura 105: Código de creación de tabla preguntas

## *Tabla Respuestas*

Contiene las respuestas utilizadas en los juegos. Tiene un identificador único para cada respuesta, un campo donde se introduce la respuesta, y finalmente un campo tipo que hace referencia a la pregunta a la que pertenece. Podemos observar que varias respuestas pertenecen a una pregunta y como explicamos en la tabla anterior cada pregunta conoce su respuesta correcta.

```
CREATE TABLE fdi.respuestas
(
  id_respuesta integer NOT NULL,
  respuesta character varying(100) NOT NULL,
  id_pregunta integer NOT NULL,
  CONSTRAINT respuestas_pkey PRIMARY KEY (id_respuesta),
  CONSTRAINT id_pregunta_fkey FOREIGN KEY (id_pregunta)
      REFERENCES fdi.preguntas (id_pregunta) MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION
)
```

Figura 106: Código creación de tabla respuestas

Tabla 24: Ejemplos de respuestas en base de datos

| id_respuesta integer | respuesta character varying(100) | id_pregunta integer |
|---|---|---|
| 1 | Es un protocolo de encaminamiento | 1 |
| 2 | Mantiene asiganacion entre nombres de host y direcciones ip | 1 |
| 3 | Analiza el trafico de red y determina si debe seguir su paso | 1 |
| 4 | Re-escribe direcciones y puertos de un paquete | 1 |
| 5 | Es sinonimo de antivirus | 2 |
| 6 | Parte de una lista de servidores raiz de los que solo hay 13 en el mundo | 2 |

## Sección 10.2.2. Servicios REST expuestos para los Juegos

A continuación exponemos todos los servicios que se han implementado para el funcionamiento de los juegos, estos servicios se han creado para facilitar la implementación en las distintas plataformas y que todas puedan acceder a unos mismos servicios para poder acceder a la información necesaria. Son servicios implementados en REST y JSON como los implementados para la infraestructura. Se encuentran detallados en el ANEXO B.

Tabla 25: Servicios expuestos para los juegos

| Servicio | Descripción |
|---|---|

| | |
|---|---|
| Get pregunta aleatoria con respuestas | No recibe parámetros y devuelve una pregunta escogida aleatoriamente de la base de datos con todas sus respuestas indicando cual es la correcta |
| Get pregunta por tipo con respuestas | Recibe como parámetro el id del tipo de pregunta y devuelve una pregunta del tipo seleccionado con sus respuestas indicando cual es la correcta |
| Get tipos de preguntas | No recibe parámetros y devuelve una lista con los tipos de preguntas persistidos en bases de datos |

## Sección 10.2.3. Quicktest

***Objetivos del juego respecto a desarrollo:*** Es un juego realizado a modo de primer modelo sencillo, de manera que se pudiera comprobar la conexión del juego a los servidores en las tres diferentes plataformas.

***Implementación (para las diferentes plataformas)***

**Aplicación web:** A continuación se describe el modo de implementar los puntos básicos del juego:

- Conexión con backend de juegos y conexión con backend de servicios: se establece una conexión inicial con el servidor, y una vez establecida se solicita el servicio correspondiente acorde a las necesidades del momento. Las preguntas se solicitan en bloque al entrar en el juego, de manera que evitamos un retardo de inicio de conexión en la carga de cada pregunta, siempre teniendo en cuenta que al ser quicktest el número de preguntas totales es 10, muy pocas preguntas para el retardo de tiempo que mostrará el principio de la partida. Para el caso de las insignias, se realizan al terminar el juego.
- Salto entre preguntas: Con la pregunta ya almacenada en la variable del juego, evitamos un consumo de tiempo en la carga de una página posterior simplemente reescribiendo la página en la que nos encontramos, sustituyendo los valores actuales por los de la pregunta a continuación.

  Para toda respuesta se guarda si el valor es o no correcto, y se debe tener en cuenta a la hora de la reescritura para modificar ese valor a la hora de clicar sobre la respuesta, evitando que al escoger la respuesta en la misma posición de la respuesta correcta anterior la detecte como correcta en favor de la nueva posición (teniendo en cuenta que la posición puede conservarse)

- Finalización de una partida: Al finalizar una partida (o bien el tiempo restante llega a cero o bien se responden las 10 preguntas dentro del límite del tiempo) se ha implementado una página que detecta la situación, de manera que en la misma se realiza un resumen de la partida.

  En el caso de que se consigan insignias, se establece una nueva conexión con el servidor para asignar las mismas a la base de datos.

- Tiempo restante: el tiempo restante dentro del límite de los 60 segundos establecidos, estará disponible en cuanto a visualización durante toda la partida. Esto es llevado a cabo mediante JavaScript.

***Lecciones aprendidas: (reflexión sobre los problemas encontrados)***

*Problema de comunicación AJAX con dominios externos:* En los prototipos que elaboramos al comienzo de curso para comprobar la correcta comunicación entre la infraestructura principal y las aplicaciones nos encontramos con un problema a la hora de establecer la conexión con el servidor por parte de nuestra aplicación web.

Realizábamos peticiones AJAX desde nuestra página web alojada en nuestro servidor local para solicitar un recurso que se encontraba en el servidor de la universidad donde está alojada toda nuestra infraestructura.

Recibíamos un fallo, pero finalmente, descubrimos que esto era debido a que no se puede invocar con AJAX un dominio externo distinto que no sea el que lance el código, en este caso el servidor de la Complutense. Pudimos continuar con las pruebas una vez que la página con la que estábamos trabajando estuviese también alojada en el servidor externo, o trabajando desde un entorno local.

**Aplicación Android:**

Tuvimos que plantear la iteración normal de la aplicación, y decidir la forma en que se sucederían las preguntas. Por sostenibilidad, decidimos no cargar todas las preguntas inicialmente, sino ir solicitándolas una a una a la Base de Datos.

Para calcular el paso del tiempo usamos contadores, y *firmamos* cada respuesta del usuario con un timestamp que luego procesamos para calcular las insignias.

***Lecciones aprendidas: (reflexión sobre los problemas encontrados)***

Con el QuickTest demostramos la viabilidad del proyecto, conectamos el dispositivo a nuestro sistema, y al servidor/servicio de preguntas, y permitimos al usuario interactuar, y ganar insignias.

En relación a desarrollo en Android aprendimos la arquitectura de la plataforma y el ciclo de vida de las aplicaciones. Descubrimos el modo en que funcionan los Activities, a usar Fragments, y a asegurar una comunicación entre ambos.

También tuvimos que aprender a usar contadores de tiempo y a manejar eventos.

Comenzamos a desarrollar QuickTest para Android desde Eclipse; pero la tendencia era que esto cambiara, y los desarrollos se llevaran a cabo desde el framework Android Studio.

El hecho de que no existiera en ese momento una versión final del framework Android Studio no fue muy conveniente. Se encontraba por entonces en una versión Beta 0.8. Aunque estable, todavía no existía una gran documentación al respecto, la cual estaba toda todavía orientada al desarrollo en Eclipse. Esto dificultó el desarrollo inicial debido a que no toda la información era extrapolable a Android Studio, y además, la herramienta traía nuevas funcionalidades, como es el caso del Gradle; un mecanismo que ayuda a los desarrolladores a hacer frente a la enorme diversidad de dispositivos, y específicamente, tamaños de pantalla; que traían de cabeza a los diseñadores gráficos, quienes tenían que configurar diferentes disposiciones de elementos, y utilizar componentes de distinto tamaño para lograr que sus aplicaciones se visualizaran correctamente en el mayor número de máquinas posible.

**Aplicación Pesada:** El entorno (IDE) escogido y utilizado finalmente para llevar a cabo la implementación y el desarrollo de los juegos en Java ha sido Eclipse. Es un entorno de desarrollo integrado multiplataforma y de código abierto que se caracteriza entre otros motivos por ser un entorno ligero (Slideshare, 2013). Esto es debido a que en un principio consta únicamente de la funcionalidad básica, pudiendo extenderla uno mismo

si es necesario mediante la instalación de diversos plugins o módulos (este ha sido el caso del módulo WindowBuilder que luego explicaremos).

El **objetivo principal del Quicktest** fue poner en práctica el formato JSON que íbamos a utilizar en el intercambio de datos de los juegos y comprobar más a fondo el funcionamiento de todos los servicios.

Puntos importantes de la implementación:

- Cabe destacar que en éste y en los demás juegos Java hemos implementado el **patrón singleton,** de esta manera podemos asegurarnos de que únicamente exista una instancia activa del usuario que ha comenzado la partida. También podemos realizar un acceso a dicho usuario desde cualquier clase de la aplicación cuando sea necesario.
- Se hace una llamada al servicio *preguntaAleatoria*, que nos devuelve una pregunta aleatoria cada vez que el usuario responde a la pregunta correctamente. En total se hará un número máximo de 10 llamadas si el tiempo de juego no ha llegado antes a su fin.
- Implementación de una cuenta atrás en el evento lanzado al comienzo del juego (perteneciente a la clase **Timer** de la biblioteca gráfica **Javax.Swing**).
- El juego finalizará cuando esta cuenta regresiva llegue a cero o el jugador haya contestado a todas las preguntas. Únicamente se mostrarán las insignias obtenidas y desbloqueadas en la partida actual.

***Lecciones aprendidas: (reflexión sobre los problemas encontrados)***

1. Realizar llamadas al servidor correctamente y aparte de utilizar los servicios pertenecientes al *backend OlimpiadaFDI*, como son el servicio de login de usuario y registro de insignias, también manejar el conjunto de servicios de preguntas relativo al *backend de juegos*.

2. Para que haya sido posible tanto el correcto funcionamiento de los servicios como de la creación de estructuras JSON fueron finalmente necesarias las siguientes librerías:
   - ✓ **Httpclient-4.3.6.jar**

   Necesaria para la construcción de aplicaciones en las que se necesite hacer una conexión a través del protocolo HTTP. En este caso, las aplicaciones son el cliente que requieren unos servicios web.
   Para hacer uso de esta librería necesitamos también las librerías Httpmime-4.3.6.jar, Fluent-hc-4.3.6.jar y Httpcore-4.3.3.jar entre otras.
   - ✓ **Org.json-20120521.jar**

   Librería que nos permitirá parsear y crear JSON.

   (En primer lugar y a modo de prueba en las primeras versiones de este juego, se importaron las clases: java.net.HttpURLConnection, Java.net.MalformedURLException y java.net.URL, así como el paquete java.io. La funcionalidad conseguida era la misma pero en este caso recibíamos y tratábamos los datos obtenidos como si fueran un String (cadena de caracteres) y no en formato JSON como finalmente acabamos utilizando).

3. Manejo de la clase **Timer** (Biblioteca gráfica Javax.Swing).

## Sección 10.2.4. Trivial

***Objetivos del juego respecto a desarrollo:*** El objetivo al desarrollar este juego era el de desarrollar un primer juego que mostrase ciertas características reales en cuanto a jugabilidad, de manera que el sistema se acerque a las plataformas convencionales en dicho aspecto. En cuanto a desarrollo de programación, uno de los objetivos era la interacción de imágenes tanto de manera estática como dinámica (respecto a funcionalidad, las imágenes que se muestran no generarán movimiento).

Además, supone una interacción continua con el servidor al realizarse continuas llamadas a sus servicios para obtener las preguntas. Así, se ha podido comprobar que el servidor soporta algo más de carga y no solo llamadas aisladas a los servicios.

***Implementación (para las diferentes plataformas)***

**Aplicación web:**

- *Imágenes:* Para la implementación de este juego, es necesario incluir imágenes. Para la realización de estas, tras documentación, se decide abordar su implementación con el uso de canvas. A modo de primeros modelos y aun desconociendo todas las funcionalidades ofrecidas por el canvas, se utiliza para dibujar las primeras imágenes del juego (el dado).
- *Imágenes dinámicas:* Donde sí obtuvimos una ventaja en el uso de canvas respecto a img, que nos permitía el dibujo de las imágenes de manera estática, es al añadir dinamismo sobre las imágenes. El uso de canvas se hace necesario para mostrar el resultado del lanzamiento del dado, así como las casillas que serán accesibles.
- *Casillas:* Al no ser las posiciones continuas, dado que se encuentran bifurcaciones en los caminos que puede seguir la ficha al lanzar el dado, nos enfrentamos a una decisión de diseño en la que podemos decidir entre mantener distintos arrays accesibles de casillas o un solo array con todas las casillas, manteniendo la lógica del programa acorde a los saltos. Se decide tomar la última implementación por evitar un array de una casilla única (casilla central) o que varios arrays compartieran la casilla central.
- *Saltos de pantalla:* Continuamente se salta entre la pantalla de preguntas y el tablero, para lo cual es necesario mostrar la puntuación que tenemos así como mantener los quesos que hemos obtenido. La manera de solucionar el paso de parámetros queda reflejada en el apartado de lecciones aprendidas.

***Lecciones aprendidas: (reflexión sobre los problemas encontrados)***

*Paso de variables entre JavaScript y php:* Es necesario mostrar en todo momento la puntuación obtenida, así como los quesos que hemos obtenido. Para pasar parámetros entre las pantallas se intercambian con un post en php. Pero necesitamos obtener dichos parámetros tanto en JavaScript como php. Para la lectura en JavaScript de un dato php es simple, con un *echo* a la variable se puede obtener. Sin embargo, en el caso contrario no es tan sencillo. Como ya se ha definido, JavaScript tiene como objetivo mantener la funcionalidad interna de la web. Así, para recoger los parámetros en php que han sido modificados en JavaScript, no se puede hacer una llamada directamente al parámetro JavaScript.

Uno de las maneras de conseguir los parámetros en php es leerlos del post que ha hecho un formulario. Afortunadamente, por JavaScript se puede no solo modificar los valores de un formulario sino además hacer submit sobre el mismo, lo cual permite que, en la siguiente pantalla, se puedan almacenar estos valores en las variables php.

Evidentemente, esto supone la inserción de un formulario, que en teoría es visible desde la página en la que se encuentre, sin embargo, por css, se puede evitar que se muestre el formulario.

**Aplicación Pesada:**

Una vez comprobado el funcionamiento de todos los servicios, este juego está destinado a ofrecernos una funcionalidad más amplia así como un interfaz de usuario más visible y elaborado.

Al contener una mayor funcionalidad, el número de clases y la complejidad del código aumentaron. Por ello fue necesario y recomendable emplear una arquitectura software de diseño más aclarativa y ordenada, basada en el modelo vista controlador (MVC), y que por lo tanto fuese capaz de separar todo aquello que se corresponde con las vistas de la lógica general que le da funcionalidad al juego.

Puntos importantes de la implementación:

- Construcción y manejo de eventos en interfaces.
- Construcción del bucle y algoritmo de juego.
- Creación de un "tablero virtual" conservando las distintas propiedades de cada casilla. (Recorrido del ArrayList de casillas mediante el uso del *patrón Iterator*).
- Mantener una sincronización entre ventanas a la hora de tirar los dados, escoger la casilla destino, contestar a las preguntas, mantener actualizado un feedback de puntuaciones, representar correctamente los quesos obtenidos hasta el momento, etc.

***Lecciones aprendidas: (reflexión sobre los problemas encontrados)***

Para la parte gráfica hemos utilizado el plugin de Eclipse denominado **WindowBuilder**. Este módulo está destinado a la creación de interfaces gráficas y para ello hace uso de una serie de elementos llamados *widgets (*que están implementados por el propio IDE) y además, de un gestor visual que utiliza la biblioteca gráfica **Swing**. Swing surgió con la necesidad de solventar el problema de portabilidad y como mejora gráfica general de su antecesor **AWT** (Franco, J. ,2014). Su utilización nos permitirá llevar a cabo la creación de ventanas con todos sus posibles componentes.

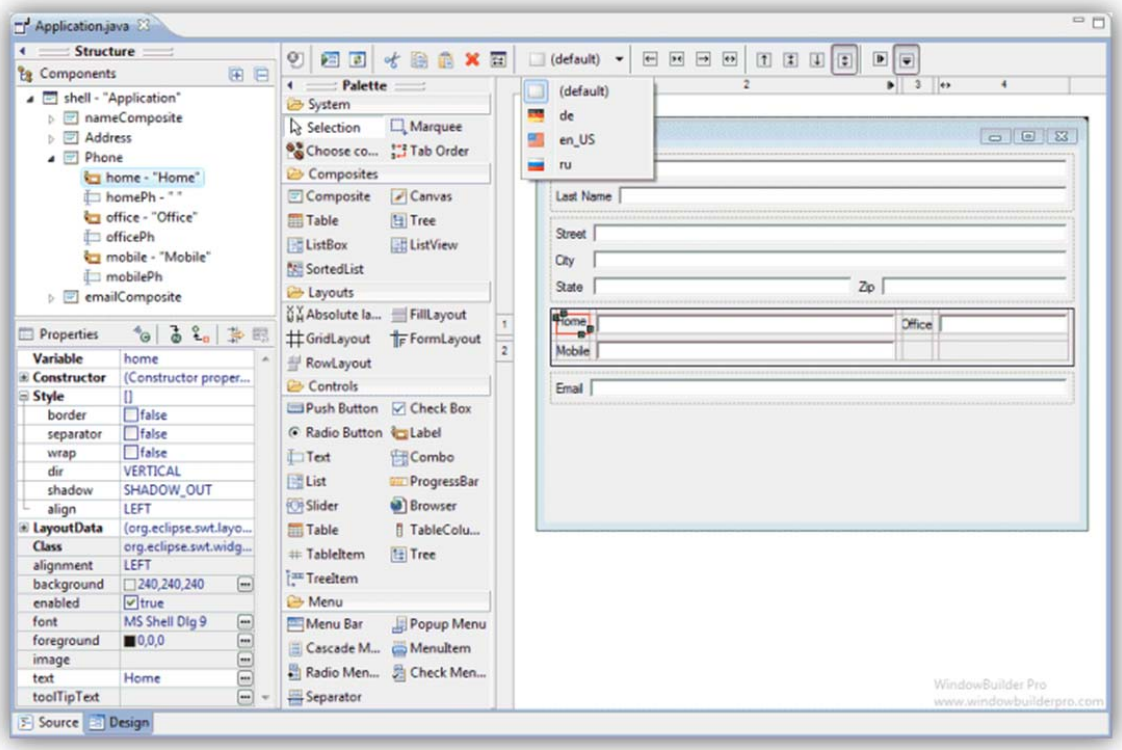



**Figura 107: Plugin WindowBuilder para Eclipse (Fuente: http://www.eclipse.org/windowbuilder/)**

## Sección 10.2.5. Hacker boy

***Objetivos de implementación:*** El objetivo inicial es disponer de un juego original que pudiese mantener una carga suficientemente alta de trabajo para el servidor y permitiese al desarrollador aprender cómo mantener el estado del sistema tras varios cambios de pantalla.

***Implementación***

- *Imágenes superpuestas:* Como novedad respecto a otros juegos, en Hacker Boy era necesaria la inclusión de varias imágenes superpuestas. Para la implementación, se ha utilizado la herramienta canvas de HTML5 (que se ha descrito en el capítulo 3), de manera que se ha desarrollado de forma más extensa que en juegos anteriores, como pudiera ser el trivial.

***Lecciones aprendidas: (reflexión sobre los problemas encontrados)***

Para dibujar imágenes diferentes dentro de un mismo canvas, se obtenían diversos problemas. En primer lugar, la colocación de las imágenes superpuestas, en la que obviamente se necesita diferenciar cual se coloca en el plano superior. Para solucionarlo, simplemente se realizaron los diferentes dibujos que se colocaron en plano superior en último lugar.

En ocasiones nos encontramos con que solo se dibujaba la primera de las imágenes sin dibujar ninguna de las otras imágenes. Para solucionar el problema, declaramos un array de las imágenes a dibujar, ya que por separado en ocasiones salía del script sin cargar todas las imágenes. En la carga de las imágenes, se dibujan por separado y ordenadamente.

## Sección 10.2.6. Bad Rabbits Die Hard!

***Objetivos del juego respecto al desarrollo:*** Aprender a manejar bucles internos del juego de manera que se simule movimiento. Así, el desarrollador ha podido aprender medios de movimiento que dan lugar a una visualización más acorde a los juegos desarrollados en el mercado actual. Además, es importante resaltar la importancia que han tenido las variables de sesión para mantener el estado del sistema.

***Implementación***

- Visibilidad: En la implementación de hacker boy, no parece intuitivo su uso. Muchos botones e imágenes colapsados. Sin una explicación detallada del mismo no parece que se pueda jugar entendiendo qué estamos haciendo. La solución al respecto, en Bad Rabbits Die Hard!, es la implementación de un sistema de botones, que se autodefinen en su texto, lo cual es mucho más intuitivo a nivel usuario.
- Dinamismo imágenes: Se añade un bucle de manera que a los elementos de la imagen se le añade un ligero movimiento, de manera que las imágenes no se encuentran estáticas, lo que de cara a la visibilidad del sistema parece una mejora.
- Variables de sesión: De cara a la consecución de una de las insignias, es necesario mantener los niveles que el usuario se ha pasado para poder conseguir una insignia. Para no mantenerlo en diferentes formularios continuos, se establecen variables asociadas a la sesión que se destruyen en el momento que el usuario deje de jugar y se reinicializan cuando el usuario empieza a jugar de nuevo.

***Lecciones aprendidas: (reflexión sobre los problemas encontrados)***

No se encontró ningún problema particular con respecto a las nuevas soluciones de desarrollo implementadas.

## Sección 10.2.7. CodeChallenge

***Objetivos del desarrollo:*** Crear un juego Android exclusivo, aprovechando la interacción que ofrecen los dispositivos móviles.

***Implementación***

Usamos las lecciones aprendidas en QuickTest para realizar CodeChallenge. También hace uso de contadores de tiempo, activities/fragments, y el resto de características que impone un desarrollo Android.

El mayor añadido fue una mejora visual que más tarde también incluimos en QuickTest, y el modo en que realizamos la interacción del jugador sobre una lista de elementos (las líneas del código).

***Lecciones aprendidas: (reflexión sobre los problemas encontrados)***

En este caso, la dificultad fue la de encontrar la forma de hacer intuitivo un programa de corrección de código para plataformas móviles. Nos decidimos por usar una lista de elementos (líneas) por su naturalidad que es.

Además, sabíamos de antemano que este componente permitía interacción; aunque la mayor dificultad venía por la forma de implementar dicha interacción.

Al final sobrecargamos el *handler* de la lista de elementos e incrustar la lógica de la aplicación en esa clase.

### Sección 10.2.8. Circuits

***Objetivos del desarrollo:*** Elaborar un juego en aplicación pesada que enseñe al usuario los fundamentos de circuitería; como lógica binaria y puertas lógicas.

***Implementación***

En cuanto a detalles de implementación, el juego Circuits comparte muchas de las directrices que sigue el juego del Trivial.

- Elaboración de los diez circuitos lógicos que se corresponden con los diferentes escenarios del juego.
- Construcción y manejo de eventos en interfaces gráficas.
- Construcción del bucle de juego.
- Manejo del Timer en cada uno de los cinco intentos de los diez niveles.
- Manejo de interfaces y clases abstractas.
- Mantener una sincronización entre las ventanas a la hora de cambiar los escenarios del juego y mantener actualizado el registro de puntuaciones y vidas restantes.

## Sección 10.3. Conclusiones

A lo largo de este capítulo hemos hablado de la suite de juegos implementada y todo lo necesario para su funcionamiento. Para cada juego hemos explicado los problemas que nos hemos encontrado en cada plataforma y las soluciones que hemos decido tomar. Finalmente hemos explicado como es la comunicación de los juegos con el servidor, que es lo que el servidor le aporta a los juegos y la estructura que utiliza el servidor para poder dar ese servicio. En esta última parte definimos la estructura utilizada para la base de datos y los servicios creados para que los juegos puedan acceder a la información.

# Bloque IV
# Caso de estudio y conclusiones

# Capítulo 11. Caso de estudio

Hemos hablado anteriormente del contexto en el que se ha desarrollado el proyecto así como de las funcionalidades que ofrece y cómo se ha implementado. Pero con solo esto, su utilidad y viabilidad no queda demostrada, por lo que decidimos para este capítulo realizar una simulación en la que se han involucrado usuarios reales interesados en el estudio de la informática para dar un mayor empaque y solidez al proyecto.

## Sección 11.1. Método

Los objetivos de este estudio son los siguientes:

- Desarrollar una **competición completa** entre dos equipos.
- Realizar testeo de la plataforma a nivel de **Prueba Integrada**
- Probar acceso a frontend de los usuarios
- Comprobar que pueden acceder al backend sin necesidad de acceder a la funcionalidad del frontend
- Confirmar conexión entre backend y frontend
- Revisar conexión entre juegos y ambos backend, tanto el de juegos como el de servicios.
- Constatar funcionalidad de los juegos en las tres plataformas
- Verificar que se realiza correctamente las inserción de insignias en el sistema
- Desarrollar una comparativa completa entre dos equipos
- Comprobar que los usuarios que tienen conocimientos previos sobre la informática tienen mayor facilidad de ascender en el ranking.
- Verificar que los usuarios pueden finalizar las partidas de los juegos sin necesidad de intervención por parte de los desarrolladores.
- Obtener feedback de apartados a mejorar por parte de los usuarios.

En general, el caso de estudio se realizará para verificar el correcto funcionamiento de la plataforma como conjunto (el proyecto funciona de manera global), asumiendo el correcto funcionamiento de cada componente individual (**Prueba Unitaria**).

El desarrollo a seguir será el siguiente:

Se estudiará y monitorizará un supuesto, basado en dos grupos compuestos por personas reales que competirán entre ellos durante dos horas. Tras ese periodo de tiempo se revisarán las puntuaciones de los grupos en la base de datos y se decidirá el vencedor.

## Sección 11.2. Desarrollo

Se crearán 2 grupos de estudio. Cada uno de ellos estará compuesto por 5 jugadores, los cuales harán uso de la plataforma para probar el rendimiento y el buen funcionamiento de la misma.

## Sección 11.2.1. Grupos

En primer lugar se realizará el registro de los usuarios del primer grupo. Cada usuario se encargó de registrarse individualmente.

### *Grupo Ready2Learn*

Tras la inserción de usuarios, acordamos como nombre del primer caso de estudio *Ready2Learn.* Finalmente el grupo quedó de la siguiente manera:

**Tabla 26: Grupo Ready2Learn**

| Ready2Learn | Puntuación total |
|---|---|
| Sandra | 0 |
| Laura | 0 |
| Julio | 0 |
| Pablo | 0 |
| Municio | 0 |

### *Grupo Hijos de Turing*

Con la motivación de superar al equipo rival en el ranking, dejamos que se registren los nuevos usuarios, quienes además son estudiantes de la Universidad Complutense de Madrid.

**Tabla 27: Grupo Hijos de Turing**

| Hijos de Turing | Puntuación total |
|---|---|
| Nuria | 0 |
| Luis | 0 |
| Claudia | 0 |
| Jeffer | 0 |
| Erik | 0 |

## Sección 11.2.2. Resumen de las fases

El grupo Ready2Learn empieza a jugar a los diferentes juegos durante el transcurso de la prueba. Como hemos mencionado anteriormente, este grupo está compuesto por integrantes ajenos a la facultad de Informática. Realizamos la siguiente captura del sistema transcurridas las dos horas de juego libre para el primer grupo:

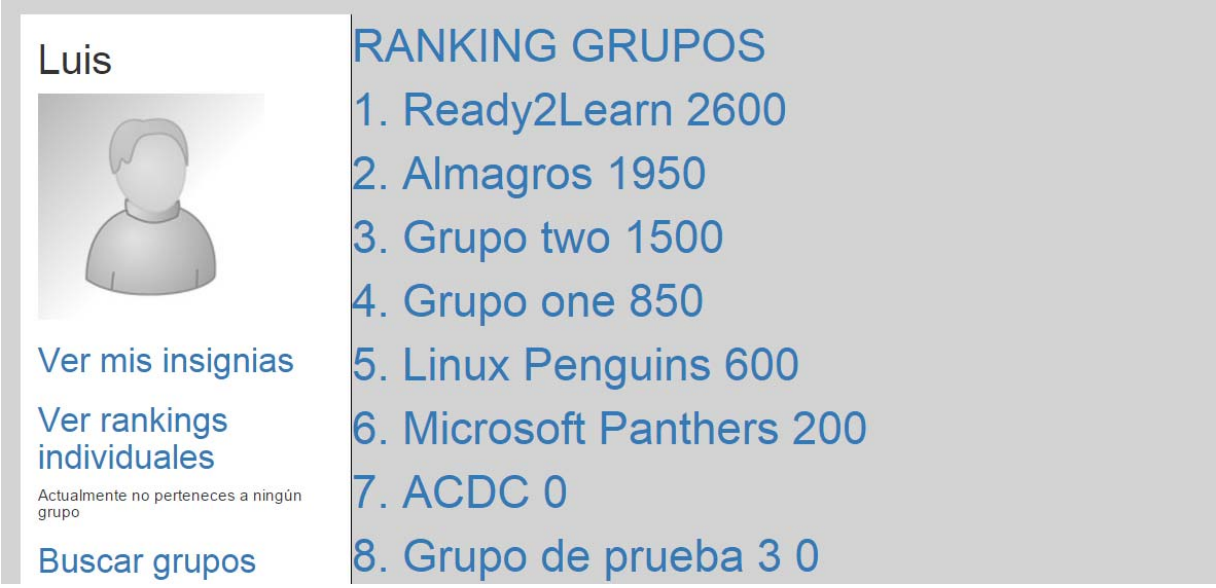



**Figura 108: Foto del sistema para Ready2Learn**

Hijos de Turing, el segundo grupo, está compuesto por estudiantes de informática y cuentan con ventaja a la hora de completar los juegos, lo que se traduce en una mayor puntuación, en rasgos generales. Tras las dos horas de juego libre, realizamos la siguiente captura del sistema:


**Figura 109: Foto del sistema para el grupo Hijos de Turing**

Estableciéndose así el grupo Hijos de Turing como claros vencedores de la competición, con una diferencia mayor a mil puntos.

### Sección 11.2.4. Discusión

En general los juegos más interactivos resultaron tener mayor éxito que aquellos que evaluaban los conocimientos técnicos en el área de la Informática.

Asumimos razonable cierto factor suerte para alguna de las insignias desbloqueadas.

El caso de estudio ha sido un éxito por los siguientes motivos:

- El sistema ha realizado cambios en su estado
- Ha mantenido la coherencia de los datos y la consistencia
- Algunos de los juegos parecieron despertar el interés de los jugadores en áreas como la circuitería y lógica, así como en el área de programación.

Del estudio o las pruebas, aunque con un margen de error grande dado que las muestras son pequeñas, se puede obtener como conclusión que los usuarios con conocimientos en informática tienen mayor facilidad a la hora de lograr los objetivos de los juegos y, acorde a ello, aumentar su medallero de insignias y puntos para las clasificaciones individuales y grupales.

Los usuarios nos dieron sus impresiones tras las pruebas, y decidimos clasificarlas entre oportunidades técnicas y funcionales. Las enunciamos con más detalle en el Capítulo 12, *Conclusiones y trabajo futuro*.

Obtenemos el feedback de los usuarios de ambos grupos y lo analizamos:

- Nos recomiendan mejoras gráficas en general.

- Valoran la fluidez de los juegos.
- Auguran un gran éxito a nuestra plataforma.
- Se interesaron por la mecánica de los juegos.

Referente a oportunidades técnicas y diseño gráfico, la mayoría de nuestros sujetos están de acuerdo en que es posible realizar una mejorar gráfica de los juegos, así como de la web.

También nos comunicaron  la carencia de una funcionalidad en la web: un botón que permita volver a la página principal. También nos recomendaron ampliar el número de tecnologías soportadas por nuestros juegos (versión para iOS) y corregir otros errores funcionales, como la opción de ver los integrantes de un grupo.

# Capítulo 12. Conclusiones y trabajo futuro

Hemos revisado a lo largo de este documento la motivación del proyecto, y la clasificación de los distintos objetivos que hemos considerado razonables.

Hemos estudiado la psicología de los juegos y su aplicación en un contexto didáctico.

Hemos realizado unos estudios previos referidos a tecnologías móviles; las plataformas más comunes, sus ventajas sobre las demás y el perfil de sus usuarios. También hemos analizado las herramientas que podemos utilizar para el desarrollo de juegos en el sistema operativo Android.

Relacionado con el desarrollo móvil, hemos estudiado también la tecnología HTML5, y los lenguajes de programación que con él se complementan, como CSS, JavaScript o PHP, y específicamente para la creación de juegos, la nueva funcionalidad en esta versión de HTML, el canvas.

También hemos estudiado una tecnología adicional que mantuviera la portabilidad y fuera multiplataforma, Java.

Por último, hemos creado los tres pilares básicos que conforman nuestro proyecto. Como hemos visto anteriormente, estos 3 pilares son:

1) Creación de una infraestructura cuya base es un **servidor de backend** que ofrece una API de servicios REST para gestionar y almacenar datos de jugadores, juegos, insignias, puntuaciones y demás.

2) Diseño y desarrollo de un **portal web** en el que se podrá visualizar los resultados fruto de la interacción de los jugadores con los juegos, incluyendo sus perfiles, insignias obtenidas, gestión de grupos, rankings de jugadores, etc.

3) Creación de una **suite de juegos** de ejemplo, desarrollados con distintas tecnologías, pero todos ellos compatibles con la API de usuarios e insignias y que ejemplifican cómo futuros desarrolladores podrán valerse para integrar sus desarrollos en nuestro sistema.

Consideramos haber cumplido todos nuestros objetivos:

- Realización de un estudio previo acerca de nuevas formas de aprendizaje.
- Realización de un estudio previo acerca de las plataformas de videojuegos existentes más exitosas hoy en día, e identificar la necesidad de una plataforma de juegos en un contexto didáctico.
- Realización de un estudio previo acerca de las tecnologías actuales: móviles, web y aplicaciones pesadas.
- Realización una plataforma basada en REST donde exponer unos servicios a los que usuarios y desarrolladores pudieran conectarse basada en las plataformas de juegos más exitosas hoy en día.
- Realización del portal web.
- Realización de dos juegos para Android, tres para Java y cuatro para el portal web didácticos.
- Demostración efectiva de la viabilidad del sistema para las plataformas Android, Web y Java.

Al margen de esto queremos hacer hincapié en la existencia de un medallero que utiliza la psicología básica de los videojuegos: esforzarse al máximo para conseguir una recompensa (en nuestro caso, insignias), y que en un entorno didáctico, incentive el afán de mejora de los jugadores.

También destacar la versatilidad de nuestra plataforma, que permite a los programadores de videojuegos publicar sus desarrollos, haciendo uso de nuestra comunidad de usuarios y aprovechar el resto de ventajas del sistema, como los servicios de insignias a través de la API pública.

## Sección 12.1. Trabajo futuro

En primer lugar, tras recibir y analizar el feedback de los usuarios de prueba, necesitamos resolver una serie de inconvenientes que se han encontrado. Algunos son errores reales, pero otros son un mal funcionamiento debido a una mala interpretación de las necesidades de los usuarios.

Relacionado con el portal web:

- Mejorar el *look & feel* de la página.
- Añadir un mecanismo para volver a la página principal, ya que en algunas secciones no hay *salida*.
- Añadir la funcionalidad de borrado de grupos cuando estos se quedan sin miembros, o eliminar un grupo de forma manual.
- Añadir la funcionalidad de ver los integrantes de un grupo.
- Perfeccionar el sistema de publicación de nuevos desarrollos y su integración en la web.
- Mejorar el componente social en la plataforma web: inclusión de un foro, chat, y mayor personalización.
- Permitir subir documentación relativa a los juegos.

Relacionado con los juegos de la suite:

- Incrementar el número de los juegos disponibles
- Mejorar el apartado gráfico de los juegos en general.
- Creación de juegos en una diversidad de plataformas y tecnologías mayor.

Estas son sólo algunas de las extensiones funcionales que se podrían realizar.

Dejamos en manos de futuros desarrolladores la posibilidad de ampliar con nuevas ideas nuestro proyecto y que su crecimiento demuestre la viabilidad comercial de nuestra plataforma.

# Chapter 13. Conclusions and future work

Along this document we have reviewed the inner motivation of the project, and the classification of the different objectives we have considered.

We have also studied the psychology of the games and its application in a didactic context.

We have delivered previous studies related to mobile technologies; most common platforms, its advantages over others, and the profile of its users. We have also analyzed the available tools for the developing of games for Android operating systems.

Related to mobile development, we have so studied the HTML5 technology, and all the programming languages that come in with it, such as CSS, JavaScript or PHP, and specifically the creation of videogames using this technology, the canvas, a feature implemented in this most recent version.

We have also studied an additional technology capable of maintaining the portability and being multiplatform, Java.

Finally, we have created the three main pillars that conform our project. As we explained before, those are:

1) Creation of an infrastructure based on a **backend server** that offers an API of REST services to manage and store our players' data, games' badges, score and so on.

2) Designing and development of a **website** in which the results of the interaction of the players would be shown, including their profiles, unlocked badges, managing groups, ranking of players, etc.

3) Creation of **suite of games** that would show off the functionality. They have been developed using different technologies, all of them compatible with the API of users and badges that will serve as an example for future developers to integrate their projects in our systems.

We consider have succeeded in all our objectives:

- Realization of a previous study related to new learning methodologies.
- Realization of a previous study related to existing most successful games' platforms.
- Realization of a previous study related to current technologies: mobile, web, and heavy applications.
- Realization of a platform based on REST on which to expose services to which users and developers could connect to, based on the most successful platforms of these days.
- Realization of the website.
- Realization of two Android games, three for Java, and four for the website.
- Demonstration of the effective viability of the system for all three platforms: Android, Web and Java.

Apart from this, we would like to emphasize the existence of the reward's mechanic, that makes use of the basic videogame's psychology: investing some effort in order to obtain some gain (in our case, badges), and which, in our learning environment, might encourage the desire for improvement in our players.

Also, highlight the versatility of our platform, which allows video games' programmers to publish their games, making use of our community of users and benefit from all the advantages of the system, such as the service of badges through the public API.

## Section 13.1. Future work

Firstly, after receiving and analyzing the proof users' feedback, we need to solve some inconveniences that have been found. Some are real errors, but others are malfunctions that are the result of a wrong interpretation of the user's needs.

Regarding the website:

- Improve the *look & feel* of the web.
- Add a mechanic to return back to the main page. In some sections there is no way of coming back.
- Add a functionality to delete groups when they lose all their members, or a way to delete them manually.
- Add a functionality to see the members of a group.
- Polish the published system of new developments and its integration in the web.
- Improve the social component of the web platform: adding a forum, chat, more personalization.
- Allow the upload of documentation related to the games.

Regarding the games:

- Increase the number of games available for the system
- Improve the graphics of the games, in general.
- Create games of a wider diversity of platforms and technologies.

These are just some of the functional extensions that could be accomplished.

We leave in the hands of future developers the possibility to improve our project and with their ideas and its growth, demonstrating the commercial viability of our platform.

# Capítulo 14. Contribución Individual

Este proyecto ha sido realizado por cuatro autores. Algunas partes se han realizado de manera grupal, pero principalmente se han separado las diferentes partes del proyecto. En este capítulo, realizaremos una revisión del trabajo realizado por cada uno

## Javier Gárate Arana

**Iteración 1 (finaliza 8/enero/2015)**

Se comenzó por la decisión de juegos a implementar y, acorde a ello, se decidió un rol para cada usuario. En el caso de Javier, se le nombró desarrollador web.

Para la documentación acerca de plataformas multijugador, y bajo el permiso y supervisión de los usuarios implicados, entrevista a diferentes usuarios expertos de dichas plataformas, que le permiten el acceso con su cuenta de manera que se pueda observar el funcionamiento de las mismas.

De manera independiente a la de Alberto, realiza un prototipo de manera que se pueda acceder al servidor por medio de aplicación web. El prototipo se limita a leer un número del 0 al 10 de manera que dicho número se reste a 10 y lo muestre por pantalla.

Pudiendo obtener información de la web (servicio REST GET, aún sin POST que requiere la formación de un JSON) desarrolla un primer modelo del quicktest, aún sin acceso a insignias ya que requieren de de servicios post.

Posteriormente y tras un estudio de cómo formar un json por medio de programación web, desarrolla un segundo modelo del quicktest, de manera que realiza peticiones a servicios del backend que requieren el envío de json al servidor.

Los siguientes juegos que se implementarán para la web, requieren del uso de un apartado visual que el quicktest no requiere. Para su implementación, estudiará las nuevas mejoras incluidas para este apartado en HTML5, y más concretamente, su nuevo elemento, canvas, estudiando tutoriales en internet así como consiguiendo acceso a libros, para el estudio del elemento, en la biblioteca de la facultad.

**Iteración 2 (finaliza 25/marzo/2015)**

Toda vez que concluye las revisiones de los juegos de la primera iteración, Javier realiza una primera aproximación al modelo de frontend. Tras la discusión con el resto del grupo acerca de cómo debería estar estructurada la misma, se desecha este primer modelo, de manera que se toma la decisión de que para el desarrollo del portal se utilizará como apoyo el framework bootstrap.

Tras un estudio del framework (documentandose en la página oficial de la web así como en cursos online) realiza un nuevo modelo de la página que incluye los primeros juegos implementados en la web aunque aún está carente de los servicios orientados a rankings y grupos, pero incluyendo servicio de registro, log in,

acceso a juegos web y vista general de la definición de los juegos, con gran parte de futuras funcionalidades de la web no implementadas pero reflejadas a modo de interfaz.

En cuanto a uso de canvas, el juego del trivial es limitado. Se realiza el desarrollo de la idea de Hacker Boy de manera que se aprovecha en mayor medida las posibilidades ofrecidas por canvas. Así, en la implementación de Hacker Boy se aprovecha el uso de canvas de manera que dentro de un mismo canvas entra más de una imagen.

**Iteración 3 (finaliza 28/mayo/2015)**

A medida que aumentaban los servicios ofrecidos por el backend orientado a servicios, se aumenta la funcionalidad del portal incluyendo gestión de rankings para usuarios individuales. Tras la inclusión de inserción en grupos, se incluye el apartado de gestión de grupos, inspirados en plataformas de servicios multijugador famosas como Xbox Live y Steam. Conjuntamente con este desarrollo se realiza la implementación de visualización de rankings de los grupos de manera que tenga sentido formar parte de alguno de los grupos.

Paralelamente al desarrollo de funcionalidad que aumenten el contenido del portal, desarrolla un nuevo juego innovador de manera que ayude a los usuarios a aprender a programar: Bad Rabbits Die Hard! Para este mismo, aprovecha en mayor medida la funcionalidad de HTML5 orientada a juegos, después de documentarse, principalmente por la web, de cómo realizar bucles para apartados repetitivos en distintos videojuegos, adaptándose para generar ligeros movimientos en los objetos dibujados, dando mayor apariencia de videojuego comercial.

Para finalizar, conjuntamente con otros, se encargó de tanto reclutar usuarios como guiarlos en el caso de estudio, aportando no solo el software sino hardware necesario para que los usuarios no tuviesen que aportar su propio hardware.

## Alejandro González Pérez

**Iteración 1 (finaliza 8/enero/2015)**

La primera tarea a realizar era conocer el servidor sobre el que la UCM nos iba a permitir trabajar. Una vez descubierto el servidor y a sabiendas que tenía un sistema operativo Linux server comencé a instalar el software que habíamos hablado que sería necesario para el proyecto.

La primera cosa que se instaló, fue el servidor de aplicaciones, un Jboss AS como explicamos en el capítulo dedicado al servidor. Se configuró para que se pudiera acceder desde fuera de la red de la UCM, exponiendo los servicios por el puerto que nos facilitaron (80).

Una vez preparado el servidor para poder desplegar nuestras aplicaciones, se instaló PostgresSQL como base de datos para poder persistir todo lo necesario relacionado con nuestro proyecto. A medida que se iba instalando el software necesario, se comienza también con la creación de los servicios a exponer para que los compañeros pudieses ir probando las llamadas desde sus distintas plataformas. Esto llevaba consigo la creación de pequeñas bases de datos para comprobar que ellos accedían a los servicios y los servicios accedían correctamente a la base de datos para formar la respuesta del servicio requerido.

Una vez realizado esto, trabajé con los compañeros para realizar correctamente las llamadas a los servicios REST desde las distintas plataformas que habíamos decidido. Cuando conseguimos acceder desde todas las plataformas correctamente, se empezaron a crear distintos servicios que respondían a la necesidad que existía en cada momento. Cuando ya conseguimos acceder desde todas las plataformas y habíamos avanzado en la definición de los juegos empecé a diseñar el esquema de la base de datos que estaría relacionada con los juegos.

Para terminar se empezó a diseñar el esquema de base de datos correspondiente a la infraestructura y la web que íbamos a exponer.

**Iteración 2 (finaliza 25/marzo/2015)**

En la segunda iteración comencé perfeccionado los servicios creados para los juegos y comprobado que la definición realizada para la base de datos de los juegos era correcta. Una vez comprobado esto se empezaron a crear los servicios REST necesarios para la infraestructura y la parte de frontend. A la vez que se avanzaban los servicios se iba completando el primer diseño realizado de base de datos para la infraestructura.

Todos los servicios se iban probando a través del SoapUi con las peticiones que realizaban los juegos ya desarrollados.

Comenzamos con la web que íbamos a mostrar y tuvimos que realizar pequeños prototipos para ver cuál encajaba más con la idea que todos teníamos. Una vez decidido esto se tomaron las decisiones necesarias y se empezó a diseñar la web. Mientras tanto se seguían realizando pequeñas correcciones de los servicios expuestos, introduciendo datos en las bases de datos y creando nuevos servicios según iban siendo necesarios.

**Iteración 3 (finaliza 28/mayo/2015)**

Para la última iteración se terminaron todos los servicios expuestos con anterioridad, comprobando su correcto funcionamiento. Se siguieron creando algunos últimos servicios necesarios para la página web y se

introdujo una buena batería de datos en base de datos.

Una vez realizado todo se comprobó a base de pruebas que todo funcionaba correctamente y se empezó a organizar y juntar toda la documentación que habíamos ido creando a lo largo del proyecto con todo lo realizado.

# Miguel Alejandro Mejía Fernández

**Iteración 1 (finaliza 8/enero/2015)**

Al principio, durante la definición del sistema, se estableció el alcance del proyecto y se comenzó a hablar sobre los juegos que tendríamos que realizar.

En el caso de Miguel, debido a que contaba con algo de experiencia en desarrollo para móviles, se le asignó la elaboración de los juegos para estos dispositivos, entre otras cosas.

Durante la primera iteración, Miguel colaboró en el diseño de la arquitectura del sistema, y cuando establecimos Android como plataforma principal, también la realización de los primeros prototipos tecnológicos:

- Una aplicación interactiva donde se mostraba el potencial de la tecnología Adobe AIR. Una serie de animaciones que saludaban al usuario y un botón para repetir el bucle. Desechamos esta tecnología debido al batacazo de Adobe Flash; y por preferir usar tecnología nativa.
- Multiply!, una prueba de concepto en Android nativo donde se accedía a la API del motor DuckDuckGo y se enviaba un número para ser multiplicado por dos. La petición de servicio era vía GET, y la respuesta en JSON debía *parsearse*.
  Más tarde se modificó este mismo prototipo para usar nuestro servicio y realizar una operación de resta.

También preparamos las herramientas de administración del sistema: acceso por túnel SSH al servidor a través de PuTTY, uso de la aplicación SoapUI para realizar llamadas REST, comprobar las rutas (endpoints) de los servicios, o el estado del servidor.

Por último, durante la primera iteración se comenzó el desarrollo del juego QuickTest.

**Iteración 2 (finaliza 25/marzo/2015)**

Durante la iteración 2, Miguel continúa puliendo el QuickTest, y comienza a indagar sobre nuevas tecnologías, centrándose, sobre todo, en el desarrollo de videojuegos.

Pronto entramos en el mundo de los frameworks especializados y los motores de juegos. Recopilamos información sobre Unity, Game Salad y Construct 2, hasta que damos con herramienta GameMaker; la cual parecía tener el equilibrio entre potencia y sencillez que buscábamos para la realización de juegos didácticos.

Realizamos varias pruebas de concepto con GameMaker: un juego y una pantalla de login. El mayor inconveniente de este motor es que está dedicado a juegos PC, y no aprovecha la capacidad de los dispositivos móviles; por ejemplo, no trae funcionalidad giroscopio, ni captura los distintos eventos derivados de la interacción táctil. Una de las mayores limitaciones que tuvimos que superar, fue el hecho de que no traía opción de teclado, algo indispensable para nuestras aplicaciones, ya que debíamos identificar al usuario antes de jugar, para poder realizar el registro de insignias.

Resolvimos esta tarea gracias a un usuario de la propia comunidad GameMaker (ClassyGoat, 2012), quien había desarrollado una aplicación (que era un juego en sí mismo) que consistía en un teclado capaz de escribir por pantalla. Había creado un botón por tecla, y las había codificado de acuerdo a la implementación interna de GameMaker; la cual no siempre se corresponde con una traducción literal (un ejemplo de esto, el

signo ‘#’ se corresponde a la tecla ‘Intro’). Hicimos las modificaciones oportunas para capturar el nombre de usuario y contraseña, y realizar un mensaje POST con los datos correspondientes, pero por limitaciones internas de GameMaker, el envío de datos JSON no era interpretado correctamente por el servidor, y recibíamos un mensaje de error. Debido esta enorme carencia, desechamos la herramienta.

**Iteración 3 (finaliza 28/mayo/2015)**

Tras la decepción que supuso GameMaker, y debido a que entrábamos en la fase final, y con tareas aún por realizar, decidimos implementar un nuevo juego, el CodeChallenge, sin más herramientas que las que proporciona Android de manera nativa.

En este aspecto, cabe destacar que la plataforma Android Studio mejoró a lo largo del primer trimestre de 2015, y que su uso para desarrollos Android ya estaba muy estandarizado.

A lo largo de esta iteración y durante Junio, nos centramos en la realización de la memoria y su traducción a inglés de los capítulos clave, a completar la documentación y a la reafirmación de nuestros conocimientos.

# Alberto Rodríguez Villalobos

El planteamiento y distribución del trabajo a lo largo del curso se puede dividir en tres fases claramente diferenciadas: Aprendizaje y análisis del funcionamiento de nuestra infraestructura así como de algunas tecnologías y nuevas herramientas de desarrollo, elaboración e implementación de juegos para la *suite* y finalización de la memoria junto con mis compañeros.

**Iteración 1 (finaliza 8/enero/2015)**

En el momento que comenzó el curso, me dediqué durante una pequeña temporada al estudio y repaso tanto de las nuevas tecnologías que emplearíamos, como de aquellas que no había manejado demasiado hasta el momento. Además, me interesé por aprender cómo funcionaba toda la parte dedicada al servidor y cómo poder acceder a ella, parte de la que se encargaba Alejandro y quién me guió durante dicho proceso.

En primer lugar, saber establecer la conexión con el servidor alojado en la Complutense desde mi equipo, ya sea mediante una conexión SSH por consola o desde el gestor de archivos Nautilus de Ubuntu (Linux). Una vez establecida la conexión, acceder al servidor de aplicaciones donde se encuentran todos los servicios REST expuestos así como modificar y/o añadir otros nuevos a los ya existentes.

Durante la realización de este proceso tuve contacto con algunas herramientas nuevas:

- Instalación del JBoss Developer Studio (IDE) para simular el comportamiento del servidor principal pero trabajando en local y así poder realizar pruebas antes de que éstas fuesen definitivas y finalmente subidas al original. Sobre el servidor local desplegar el mismo servidor de aplicaciones que en la máquina virtual principal, JBoss AS 7.1.1. Configuración de la herramienta software Maven.
- Instalación de la aplicación SOAPUI para poder probar todos los servicios expuestos.
- Instalación del gestor de B.D PostgreSQL, pgAdmin, su configuración y conexión, para así a lo largo de la etapa de implementación poder insertar las preguntas referidas a los juegos en su base de datos correspondiente.

Adicionalmente, me familiaricé con el formato JSON (W3schools, 2000) ya que esa sería la estructura de datos que introduciríamos en las URLs destinadas al funcionamiento de los servicios REST.

Paso siguiente fue el de crear una breve implementación capaz de conectar los futuros juegos con el servidor a la hora de hacer peticiones a los servicios expuestos. Para ello, realicé dos prototipos tecnológicos (pequeña aplicación cuyo código ya incluía la formación de estructuras JSON) cuyo único objetivo sería el de conseguir establecer dicha comunicación.

Estos prototipos Web y Java nos servirían a la hora de implementar los juegos que se crearían con dichas tecnologías.

Para finalizar este estudio y con los conocimientos adquiridos, subí el prototipo web al servidor, pudiéndose visualizar y probar en la siguiente dirección http://ssii2014.e-ucm.es/PrototipoHTML/pages/multiplica2.html
.

A continuación, comenzó la etapa destinada a la creación de la suite de juegos, en la que cada uno se centró en un lenguaje de programación distinto a la hora de la implementación. En mi caso, me centré en los juegos orientados al lenguaje Java.

En total se han creado un total de 3 juegos en Java. En primer lugar, el juego multiplataforma Quicktest (que nos serviría para poner en práctica los servicios creados, tanto los del backend OlimpiadaFDI como los del correspondiente a los juegos), a continuación el Trivial y por último Circuits, un juego en el que se ponen a prueba los conocimientos sobre circuitos lógicos y la capacidad de respuesta rápida del jugador.

Antes de la finalización de esta iteración, comenzamos con la implementación del primero de ellos, el Quicktest.

**Iteración 2 (finaliza 25/marzo/2015)**

En cursos anteriores, aprendimos algunos conocimientos sobre cómo realizar pequeñas ventanas simples en Java, pero necesitábamos ir un poco más allá y dejando el Quicktest a un lado, los demás juegos ya requerían una interfaz mucho más elaborada y por tanto los Layouts dejarían de ser tan simples. Aquí entra en juego el aprendizaje de la biblioteca gráfica **Swing**, con la que la creación de las vistas de la aplicación se hizo más elaborada e interesante. Esto fue posible gracias a la visibilidad y versatilidad del plugin **WindowBuilder**.

Esta iteración la dediqué a la finalización del juego Quicktest y a realizar el Trivial. Con los conocimientos adquiridos en la primera etapa, pude insertar las preguntas de las diferentes asignaturas en su correspondiente Base de Datos.

**Iteración 3 (finaliza 28/mayo/2015)**

Esta última iteración fue dedicada a la finalización del juego Circuits y de la Memoria. Una vez que lo juegos de todas las plataformas estaban disponibles y funcionaban correctamente, realizamos un pequeño caso de estudio del que pudimos obtener un valioso feedback por parte de los usuarios encuestados, tanto para la mejora de los juegos como del portal en el que se visualizan todas sus puntuaciones.

# Chapter 15. Personal contribution

This project has been developed by four authors. Some parts have been in group way, but mainly the different parts of the project have been split. In this chapter, we will carry out a review of the work carried out by each member of the team.

## Javier Gárate Arana

**Iteration 1 (ends 8/January/2015)**

The team started with the decision which games to implement and, according to this, it was decided a role for each team member. In the case of Javier, he was appointed web developer.

For documentation about multiplayer platforms, and under the permission and supervision of involved users, interviewed different expert users on these platforms, allowing him to access their own account so that he could can observe the platform's operation.

Independently from Alberto, Javier performs a prototype so you can access the server through web application. The prototype is limited to read a number from 0 to 10 so that number will subtract 10 and display it on the screen.

You may obtain information from the web (REST service GET, even without POST that requires the formation of a JSON) which develops a first model of the quicktest, even without access to badges since they require of service post.

Subsequently, after a study of how to build a json through web programming, he developed a second model of the quicktest, so it made requests to the backend services that require sending json to the server.

The following games which will be implemented for the web, will require the use of a visual section that the quicktest does not require. For the implementation, it will consider further improvements included for this section in HTML5, and more specifically, its new element, canvas, studying tutorials on internet as well as getting access to books, for the study of the element, in the faculty's library.

**Iteration 2 (ends 25/March/2015)**

Once the games' reviews concluded in first iteration, Javier made a first approach to the frontend model. After the discussion with the rest of the team regarding how frontend should be structured, this option was rejected so the decision to use framework bootstrap, to develop the web portal, was taken..

After a study of the framework (looking at the official website of the web as well as online courses) a new model of the web was carried out, which includes the first games implemented although still lacks of services for rankings and groups, but includes registration, log in, access to web games and general view of the game's definition, already with future functionalities of the web not yet implemented but reflected as an interface.

Regarding use of canvas, the trivial game is limited. The development of the idea of Hacker Boy is used which benefits from the possibilities offered by canvas. Thus, in the implementation of Hacker Boy canvas, gets more than one image within the same canvas.

**Iteration 3 (ends 28/may/2015)**

As the number of services offered by the backend are increased, so does the functionality of the portal including the management of rankings for individual users. After the inclusion of groups, group management is included, inspired by famous Steam and Xbox Live multiplayer service platforms. Along with this development is the implementation of the visualization of rankings of groups, so it does make sense joining any of them.

In parallel to the development of functionality that will increase the portal's content, develops a new innovative game so help users learn how to program: Bad Rabbits Die Hard! For this game he takes the advantage of game-oriented HTML5 functionality, after gathering information, primarily from the web, on how to perform loops for repetitive sections in different video games, adapting itself to generate light movements on the drawing objects, giving greater appearance of commercial video game.

Finally, together with other members of the team, was responsible for both recruiting and coaching users for the case study, providing not only the software but hardware required so that users didn’t have to use their own devices.

## Alejandro González Pérez

**Iteration 1 (ends 8/January/2015)**

The first task at hand was to find the server on which the UCM would allow us to work on. Once discovered the server and knowingly having a Linux operating system server started to install software that we talked to that it would be necessary for the project.

The first thing that was installed, was the application, a Jboss AS server as explained in the chapter dedicated to the server. It was configured so that he could access from outside the network of the UCM, exposing services through the port that we facilitated (80).

Once prepared the server to deploy our applications, settled PostgresSQL as the database to be able to persist everything related to our project. As to the necessary software was installed, begins also with the creation of services to expose so that colleagues could try calls from their various platforms. This carried with him the creation of small databases to verify that they are accessing services and services properly accessing the database to form the required service response.

Once done, I worked with colleagues for correctly calling REST services from different platforms that we had decided. When we get access from all platforms correctly, it began the creation of various services that responded to the need that appeared in every moment. When we already get access from all platforms and we had advanced in the definition of the games I started to design the schema of the database that would be associated with the games.

Finally began to design the base data schema corresponding to infrastructure and the web that we were going to expose.

**Iteration 2 (ends 25/March/2015)**

On the second iteration started fine tuning the services created for games and found that the definition for the database of the games was correct. Once verified this, began to create the necessary REST services for infrastructure and the frontend part. At the same time services are advancing, I was completing the first design of data base for the infrastructure.

All services were tested through the SoapUi with requests that performed the games already developed.

We started with the web that we were going to show and had to create small prototypes to see which would fit the best the idea that we all had. Once decided that, the necessary decisions were taken and I began to design the website. Meanwhile still being small corrections of the services set forth, entering data in the databases and creating new services as they were still necessary.

**Iteration 3 (ends 28/may/2015)**

For the latest iteration we terminated all services stated above and checked its correct operation. I followed establishing some past services for the website and introduced a good battery of data in the database.

Once all found to be based on evidence, and everything worked properly; we began to organize and put together all the documentation that we had been building throughout the project with all what has been done.

# Miguel Alejandro Mejía Fernández

**Iteration 1 (ends 8/January/2015)**

At the beginning, during the definition of the system,  the scope of the project was established and we began talking about the games we'd have to make.

In Miguel´s case, since it had some experience developing for mobile, was assigned the preparation of the games for these devices, among other things.

During the first iteration, Miguel contributed to the design of the system architecture, and when Android was established as main mobile platform, also the realization of the first technological prototypes:

- An interactive application which showed the potential of Adobe AIR technology. A series of animations that greeted the user and a button to repeat the loop. We rejected this technology due to the thud of Adobe Flash; and prefer to use native technology.
- Multiply!, a proof of concept in native Android which accessed the search engine DuckDuckGo API and could send a number to be multiplied by two. The service request was via GET, and JSON response had to be parsed.

  Later on, this same prototype was modified to use our service to perform a subtract operation.

Also prepare system management tools: SSH tunnel access to the server through PuTTY, use of SoapUI for making REST calls, check the paths (endpoints) services, or the status of the server.

Finally, during the first iteration is began the development of the game Quicktest.

**Iteration 2 (ends 25/March/2015)**

During the iteration 2, Miguel continues polishing the QuickTest, and begins to inquire about new technologies, focusing, above all, on the development of video games.

We soon entered the world of specialized frameworks and games engines. We collect information on Unity, Game Salad and Construct 2, until we hit the tool GameMaker; which seemed to have the balance between power and simplicity we wanted to carry out educational games.

We do several proof of concepts with GameMaker: a game and a login screen. The major drawback of this engine is that it is devoted to PC games, and not leveraging the capacity of mobile devices; for example, does not bring functionality gyroscope, or captures the various events arising from the touch interaction. One of the major constraints that we had to overcome, was the fact that brought no option of keyboard, indispensable for our applications, since we should identify the user before you play, in order to make the registration of badges.

We solved this task thanks to a user of the community itself GameMaker (ClassyGoat, 2012), who had developed an application (that was a game in itself) which consisted of a keyboard capable of writing for the screen. It had created a button by button, and had encoded them according to the internal implementation of GameMaker; which do not always correspond to a literal translation (an example of this, the sign '#' corresponds to an “Enter”)

We made the appropriate modifications to capture the user name and password, and make a POST message with the corresponding data, but by internal limitations of GameMaker, sending JSON data was not correctly interpreted by the server, and we received an error message. Because of this huge lack, we throw away the tool.

**Iteration 3 (ends 28/may/2015)**

After the disappointment that was GameMaker, and because we entered the final phase, and with tasks still ahead, decided to implement a new game, the CodeChallenge, without further tools than those provided by Android, natively.

In this respect, notably that the platform Android Studio improved during the first quarter of 2015, and that its application for Android development was already very standardized.

Along this iteration and during June, we focused on the preparation of documentation for the memory and its translation into English, and reaffirmation of our knowledge.

# Alberto Rodríguez Villalobos

The planning and distribution of work throughout the year can be divided into three distinct phases: learning and analysis of the performance of our infrastructure as well as some technologies and new development tools, elaboration and implementation of games for the suite and completion of the memory together with my colleagues.

**Iteration 1 (ends 8/January/2015)**

When the course started, I spent some little time to study and review new technologies, some of which I had not handled too far. In addition to this, I was interested to learn how it worked the entire dedicated server and how to access it, part of which Alejandro, who was responsible for that, guide me during such process.

First of all, know the connection with the server hosted in the Complutense University from my computer, either using a SSH connection or from Nautilus file manager, in Ubuntu. Once the connection was established, access to the application server where all the exposed REST services are, as well as modify and/or add new ones to the existing ones.

During this process I had contact with some new tools:

- Installation JBoss Developer Studio (IDE) to simulate the behaviour of the main server but working in local and thus perform tests until they were finally rised to the original and definitive. On the local server deploy the same application server than the one in the main virtual machine, JBoss AS 7.1.1. The configuration of the software Maven.
- Installation of application SOAPUI to try all services exposed.
- Installing B.D PostgreSQL, pgAdmin, its configuration and connection manager, to insert the questions referring to the games on its corresponding database as well throughout the implementation phase.

In addition, I became familiar with the format JSON (w3schools, 2000) since that would be the data structure that we would introduce in the URLs to the functioning of the REST services.

Next step would be finally to create a brief implementation capable of connecting the future games with the server when making requests to the exposed services. So did two technological prototypes (small application whose code already included the formation of JSON structures) whose sole objective would be to establish this communication.

These Web and Java prototypes shall serve us to implement the games that would be created with these technologies.

To complete this study and with the acquired knowledge, got the prototype web server, being able to visualize and test at the following address

http://ssii2014.e-ucm.es/PrototipoHTML/pages/multiplica2.html

Then began the stage aimed at the creation of the suite of games, in which each focused mostly on a different programming language at the time of implementation. In my case, I focused on the language-oriented games in Java.

We have created a total of 3 games in Java. Firstly, the multiplatform game Quicktest (that we would put into practice the services created, both the OlimpiadaFDI backend and those of the corresponding games), then the Trivial and finally Circuits, a game in which are tested knowledge of logic circuits and rapid responsiveness of the player.

Before the end of this iteration, we started with the implementation of the first of these, the Quicktest.

**Iteration 2 (ends 25/March/2015)**

In previous years, we gained some knowledge about how to make simple small windows in Java, but needed to go a little further and leaving the Quicktest aside, other games already required a much more elaborate interface and therefore layouts would no longer be so simple. Here comes into play the learning of the **Swing** graphical library, with which the creation of views became more elaborate and interesting. This was possible thanks to the visibility and versatility of the plugin **WindowBuilder**.

This Iteration was devoted to completion of the Quicktest game and perform the Trivial. With the knowledge acquired in the first stage, I could insert the questions of the different subjects in their corresponding database.

**Iteration 3 (ends 28/may/2015)**

This latest iteration was dedicated to the completion of the game Circuits and the Memory. Once the games for all platforms were available and working properly, we carry out a small case study that we were able to obtain valuable feedback from surveyed users, both for the improvement of the games and the portal in which all their scores are displayed.

# Bibliografía y referencias


***Anandtech 2015***

Smith, R. (2015) *Valve to Showcase SteamVR Hardware, Steam Machines, & More at GDC 2015*. Disponible online en http://www.anandtech.com/show/9003/valve-to-showcase-steamvr-hardware-steam-machines-more-at-gdc-2015/ (última comprobación: julio de 2015).

***Android Developer (2015)***

Google Inc. (2015a) Dashboards documentation. Disponible online en http://developer.android.com/about/dashboards/index.html (última comprobación: julio de 2015).

***Android Studio (2015)***

Google Inc. (2015b) Android Studio Release Notes. Disponible online en https://developer.android.com/tools/revisions/studio.html (última comprobación: julio de 2015).

***Avşaroğulları, E. (2011)***

Avşaroğulları, E. (2011). Hibernate Overview. Disponible online en http://www.onlinetechvision.com/hibernate-overview/ (última comprobación: junio de 2015).

***Blog de Ameu8 (3/9/2014)***

Ameu8 (2014) Lo que gastamos en Apps. Disponible online en http://ameu8.com/aplicaciones-moviles/lo-que-gastamos-en-apps (última comprobación: julio de 2015).

***Bootstrap settings (s.f.)***

Bootstrap (s.f.). Global CSS settings, fundamental HTML elements styled and enhanced with extensible classes, and an advanced grid system. Disponible online en http://getbootstrap.com/css/ (última comprobación: julio de 2015).

***Colaboradores de la Wikipedia (s.f.)***

***Construct 2 (2015)***

Scirra Ltd. (2015). Create Games with Construct 2. Disponible online en https://www.scirra.com/construct2 (última comprobación: junio de 2015).

***Core Education (2013)***

Core Education (2013). Ten trends 2013 – 9. Ubiquitous learning. Disponible online en http://www.core-ed.org/thought-leadership/ten-trends/ten-trends-2013/ubiquitous-learning (última comprobación: julio de 2015).

***Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011).***

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness. En actas de 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek '11 (pp. 9–11).

***Education 2025 (2015)***

Education 2025 (2015). Ubiquitous Learning. Disponible online en https://education-2025.wikispaces.com/Ubiquitous+Learning (última comprobación: julio de 2015).

***Ewan Spence (1/11/2015)***

Spende, E. (2015) Why Is Nobody Using Android 5.0 Lollipop?. Disponible online en http://www.forbes.com/sites/ewanspence/2015/01/11/nobody-is-using-android-lollipop/ (última comprobación: julio de 2015).

***Franceschini, G. (2011), Web de Tecnético***

Franceschini, G. (2011). Usuarios de iOS invierten más tiempo jugando que cualquier otro usuario. Disponible online en https://www.tecnetico.com/19276/usuarios-de-ios-invierten-mas-tiempo-que-cualquier-otro-usuario/ (última comprobación: julio de 2015).

***Franco, J. (2014)***

Franco, J.D. (2014). Diferencias entre Java Swin y AWT. Disponible online en http://es.slideshare.net/extrajuandavid/diferencias-swing-y-awt (última comprobación: julio de 2015).

***GameMaker (2012)***

GameMaker Community (2012). On Screen Keyboard example. Disponible online en http://gmc.yoyogames.com/index.php?showtopic=553495 (última comprobación: julio de 2015).

***GameMaker (2015)***

Yoyo Games (2015). Game Maker Studio. Disponible online en http://www.yoyogames.com/studio. (última comprobación: julio de 2015).

***Game Salad (2015)***

Game Salad (2015). Game Salad. Disponible online en http://gamesalad.com/ (última comprobación: julio de 2015).

***Gauchat (2012)***

Gauchat, J. D. (2013) El gran libro de HTML5, CSS3 y JavaScript. Marcombo, 2012

***Gee, J. P. (2007)***

Gee, J. P. (2007). Good videogames and good learning: collected essays on video games. New York: Peter Lang Publishing.

***Hipertextual (23/4/2014)***

HiperTextual (2014a). Sistemas operativos muertos: WebOS. Disponible online en http://hipertextual.com/archivo/2014/04/webos/ (última comprobación: julio de 2015).

***Hipertextual (21/10/2014)***

HiperTextual (2014b). Sistemas operativos muertos: Symbian. Disponible online en http://hipertextual.com/archivo/2014/10/historia-symbian/ (última comprobación: julio de 2015).

***Intel, (18/3/2015)***

Intel Corp. (2015) El nuevo entorno de desarrollo de aplicaciones para Android: Android Studio (Beta). Disponible online en https://software.intel.com/es-es/articles/el-nuevo-entorno-de-desarrollo-de-aplicaciones-para-android-android-studio-beta (última comprobación: marzo de 2015).

***JSON Org. (2015)***

JSON Org (2015). Introducing JSON. Disponible online en http://www.json.org (última comprobación: julio de 2015).

***Kongregate FAQ (2015)***

Kongregate (2015). Kongregate FAQ Help: Earning Points. Disponible online en http://www.kongregate.com/pages/help#earning (última comprobación: julio de 2015).

***Lextrait, V. (2007-2013)***

Lextrait, V. (2013). The Programming Languages Beacon. Disponible online en http://www.lextrait.com/Vincent/implementations.html (última comprobación: julio de 2015).

***LibrosWeb (2015)***

LibrosWeb (2015). Introducción a XHTML, Capítulo 1 – Breve historia de HTML. Disponible online en http://librosweb.es/libro/xhtml/capitulo_1/breve_historia_de_html.html (última comprobación: julio de 2015).

***Malone, T. (1981)***

Malone, T. (1981) What makes computer games fun? Byte, 6(12), 258–276.

***Martín, V. (2013) (¡atención al número!)***

Martin, V. (2012) El perfil de los jugadores sociales. Disponible online en http://victormartinp.com/2012/01/el-perfil-de-los-jugadores-sociales-infografia/ (última comprobación: julio de 2015).

***Michael, D., & Chen, S. (2006)***

Michael, D., & Chen, S. (2006) Serious Games: Games that Educate, Train, and Inform. Boston, MA: Thomson.

***Moodle Pty Ltd (2014), Badges documentation***

Moodle Pty Ltd (2014), Badges documentation. Disponible online en https://docs.moodle.org/29/en/Badges (última comprobación: julio de 2015).

***Mozilla Foundation (2014), Mozilla Open Badges***

Mozilla Foundation (2014), Mozilla Open Badges. Disponible online en http://openbadges.org/ (última comprobación: Junio de 2015).

***Prensky, M. (2001)***

Prensky, M. (2001) Digital Game Based Learning. New York: McGraw-Hill.

***Prensky, M. (5/06/2014)***

Fominaya, C. (2014). Entrevista con el creador de los términos "nativo" e "inmigtante digital". Disponible online en http://www.abc.es/familia-educacion/20141118/abci-nativos-digitales-prensky-201410291748.html (última comprobación: julio de 2015).

***Prensky, M. (18/11/2014)***

Tiching (2014). Entrevsista con Mark Prensky "Debemos descubrir la pasión de cada estudiante". Disponible online en http://blog.tiching.com/marc-prensky-debemos-descubrir-la-pasion-de-cada-estudiante/ (última comprobación: julio de 2015).

***Puromarketing (2013)***

PuroMarketing (2013). Los usuarios invierten ya más en juegos para su móvil que para las videoconsolas. Disponible online en http://www.puromarketing.com/72/15355/usuarios-invierten-juegos-para-movil-para-videoconsolas.html (última comprobación: julio de 2015).

***Schroeder, Thüs y Amine Chatti (2014)***

Schroeder, U., Thüs, H. y Amine Chatti, M. (2014) Documentación de clase de la asignatura Advanced Learning Technologies, curso 2013-2014, Universidad RWTH Aachen, Alemania.

***Slideshare (2013)***

González Flores, M.A., Aguilar Guzmán, K.E., Sical Raxcacó, M.O. et al (2013). Presentación Eclipse. Disponible online en http://es.slideshare.net/MagaLasic/presentacion-eclipse-grupo-6 (última comprobación: julio de 2015).

***Squire, K. (2005)***

Squire, K. (2005). Changing the game: What happens when video games enter the classroom. Innovate, Journal of Online Education, 1(6).

***Tanous, J. (23/1/2012)***

Tanous, J. (2012). iSuppli Predicts Windows Phone's Edge Over iOS by 2015. Disponline en http://www.macobserver.com/tmo/article/isuppli_joins_others_in_predicting_windows_phones_edge_over_ios_by_2015 (última comprobación: julio de 2015).

***Unity (2015)***

Unity Technologies (2015). Unity – Game Engine. Disponible online en https://unity3d.com/es (última comprobación: julio de 2015).

***Valve Corporation (2015).***

Valve Corporation (2015). Preguntas y respuestas básicas sobre Steam Cromos. Disponible online en http://www.steamcromos.es/guia/preguntas-y-respuestas-basicas (última comprobación: julio de 2015).

***Whitwam, R. (3/4/2014), web ExtremeTech***

Whitman, R. (2014) Windows Phone 8.1 vs. Android 4.4 KitKat: Microsoft plays catch-up, but still lags behind. Disponible online en http://www.extremetech.com/mobile/179775-windows-phone-8-1-vs-android-4-4-kitkat-microsoft-plays-catch-up-but-still-lags-behind (última comprobación: julio de 2015).

***W3schools (2000)***

W3Schools (2000). JSON Tutorial. Disponible online en http://www.w3schools.com/json/default.asp (última comprobación: julio de 2015).

***99 points (2010)***

Rasool, Z. (2010). 50 CSS techniques and Tips which you always need.Disponible online en http://www.99points.info/2010/03/50-css-techniques-and-tips-which-you-always-need/ (última comprobación: julio de 2015).

# Anexo

## Anexo A. Documentación de los RestServices expuestos en backend de juegos

### 1) Pregunta aleatoria con sus respuestas

Method: GET

URL:

http://ssii2014.e-ucm.es:80/OlimpiadaFDIServices/rest/quickTest/preguntaAleatoria

Response:


```
{
  "code": 0,
  "message": "OK",
  "result":   {
    "pregunta":     {
      "idPregunta": 1,
      "pregunta": "Quienes pertenecen a olimpiadaFDI ?",
      "respuestaCorrecta": 3,
      "tipo": 1
    },
    "respuestas":     [
          {
      "idRespuesta": 1,
      "respuesta": "Luis,Pedro,Edu",
      "idPregunta": 1
    },
          {
      "idRespuesta": 2,
      "respuesta": "Luis,Pedro,Maria",
      "idPregunta": 1
    },
          {
      "idRespuesta": 3,
      "respuesta": "Alberto,Miguel,Javi,Alex",
      "idPregunta": 1
    },
          {
      "idRespuesta": 4,
      "respuesta": "Juanin,Luisa",
      "idPregunta": 1
    }
  ]
  }
}
```

Se realiza una petición al servicio REST que devuelve una pregunta Aleatoria. Dentro de la pregunta, en el campo respuestaCorrecta, se indica el id de la respuesta correcta

## 2) Pregunta por tipo con sus respuestas

Method: GET

URL:

http://ssii2014.e-ucm.es:80/OlimpiadaFDIServices/rest/quickTest/preguntaPorTipo/{tipo}

{tipo} ejemplo = 1

Se realiza una petición al servicio REST que devuelve una pregunta del tipo indicado en la url. Dentro de la pregunta, en el campo respuestaCorrecta, se indica el id de la respuesta correcta

Response:


```
{
  "code": 0,
  "message": "OK",
  "result":   {
    "pregunta":      {
      "idPregunta": 4,
      "pregunta": "En que mes se presenta el proyecto?",
      "respuestaCorrecta": 16,
      "tipo": 1
    },
    "respuestas":      [
            {
      "idRespuesta": 13,
      "respuesta": "Diciembre",
      "idPregunta": 4
    },
            {
      "idRespuesta": 14,
      "respuesta": "Enero",
      "idPregunta": 4
    },
            {
      "idRespuesta": 15,
      "respuesta": "Agosto",
      "idPregunta": 4
    },
            {
      "idRespuesta": 16,
      "respuesta": "Junio",
      "idPregunta": 4
    }
   ]
  }
}
```

## 3) Tipos de preguntas existentes

Method: GET

URL:

http://ssii2014.eucm.es:80/OlimpiadaFDIServices/rest/quickTest/obtenerTiposDePreguntas

Devuelve los tipos de preguntas existentes.

Response:


```
{
  "code": 0,
  "message": "OK",
  "result":    [
        {
      "idTipo": 1,
      "descripcion": "Miembros del grupo"
    },
        {
      "idTipo": 2,
      "descripcion": "Deportes"
    }
  ]
}
```

# Anexo B. Documentación de los RestServices expuestos en backend de servicios

## 1) RestService InsertarInsignia

Method: POST

Consumes: application/json

URL:

http://ssii2014.e-ucm.es:80//OlimpiadaFDIServices/rest/insignias/insertarInsignia

Este servicio recibe un JSON con varios campos, pertenecientes a una insignia, y los introduce en la base de datos de insignias, de esta manera creamos una insignia con sus datos

Request:

```
{
  "idInsignia": 2,
  "descCorta": "Mi segunda",
  "descLarga": "Mi segunda Insignia",
  "puntuacion": 2
}
```

Response:

```
{
  "code": 0,
  "message": "OK",
  "result": null
}
```

## 2) Rest Service Insertar Usuario

Method: POST

Consumes: application/json

URL:

http://ssii2014.e-ucm.es:80//OlimpiadaFDIServices/rest/insignias/insertarUsuario

Este servicio recibe un JSON con varios campos, pertenecientes a un usuario, y los introduce en la base de datos de usuarios, de esta manera creamos un usuario con sus datos y contraseña

Request:

```
{
  "nombre": "User1",
  "correo": "user1@ucm.es",
  "pass": "pass1"
}
```

Response:

```
{
  "code": 0,
  "message": "OK",
  "result": null
}
```

## 3) RestService InsertarGrupo

Method: POST

Consumes: application/json

URL:

http://ssii2014.e-ucm.es:80/OlimpiadaFDIServices/rest/insignias/insertarGrupo

Este servicio recibe un JSON con varios campos, pertenecientes a un grupo, y los introduce en la base de datos de grupos, de esta manera creamos un grupo con sus datos

Request:

```
{
  "idGrupo": 2,
  "descripcion": "Grupo two"
}
```

Response:

```
{
  "code": 0,
  "message": "OK",
  "result": null
}
```

## 4) RestService AsignarInsigniaAUsuario

Method: POST

Consumes: application/json

URL:

http://ssii2014.e-ucm.es:80//OlimpiadaFDIServices/rest/insignias/asignarInsigniaAUsuario

Este servicio recibe un JSON con un idUsuario y un idInsignia (que ambos existan en sus respectivas tablas) e introduce un registro en UsuarioHasInsignias que nos dice que usuario tiene que insignias

Request:

```
{
  "nombreUsuario": "User1",
  "idInsignia": 2
}
```

La respuesta depende de. Si el usuario no tenía la insignia en el momento de llamar el servicio, responde con code=0. Por otro lado, si el usuario ya tenía la insignia, devuelve un code=2

Response al *desbloquear* la insignia:

```
{
  "code": 0,
  "message": "OK",
  "result": null
}
```

Response si ya se había desbloqueado la insignia:

```
{
  "code": 2,
  "message": "",
  "result": null
}
```

## 5) RestService AsignarUsuarioAGrupo

Method: POST

Consumes: application/json

URL:

http://ssii2014.e-ucm.es:80//OlimpiadaFDIServices/rest/insignias/asignarUsuarioAGrupo

Este servicio recibe un JSON con un idUsuario y un idGrupo (que ambos existan en sus respectivas tablas) e introduce un registro en GrupoUsuarios que nos dice que Grupo tiene que usuarios

Request:

```
{
  "nombreUsuario": "User1",
  "idGrupo": 2
}
```

Response:

```
{
   "code": 0,
   "message": "OK",
   "result": null
}
```

## 6) RestService getInsigniasUsuario

Method: GET

URL:

http://ssii2014.e-ucm.es:80//OlimpiadaFDIServices/rest/insignias/insigniasUsuario/{usuario}

{usuario} ejemplo = alberto

Este servicio recibe por la url directamente el nombre del usuario del cual queremos conocer sus insignias, se encarga de buscar el id relacionado con ese nombre (que debe ser único) y devuelve las insignias asociadas al usuario

Response:


```
{
  "code": 0,
  "message": "OK",
  "result":   {
    "usuario": "alberto",
    "insignias":      [
            {
        "idInsignia": 2,
        "descCorta": "Mi segunda",
        "descLarga": "Mi segunda Insignia",
        "puntuacion": 2
      },
            {
        "idInsignia": 1,
        "descCorta": "Mi primera",
        "descLarga": "Mi primera Insignia",
        "puntuacion": 1
      }
    ]
  }
}
```

## 7) RestService logginUsuario

Method: POST

Consumes: application/json

URL:

http://ssii2014.e-ucm.es:80//OlimpiadaFDIServices/rest/insignias/logginUsuario

Este servicio recibe un JSON con un nombre de usuario y pass de usuario y comprueba contra base de datos que exista el usuario y que el pass sea correcto (hace un loggin)

Request:

```
{
  "nombre": "User1",
  "pass": "pass1"
}
```

Response:

```
{
  "code": 0,
  "message": "OK",
  "result": "loggin correcto"
}
```

O los posibles errores

```
{
  "code": -1,
  "message": "OK",
  "result": "El usuario no existe"
}
```

```
{
  "code": -1,
  "message": "OK",
  "result": "El password es incorrecto"
}
```

## 8) RestService getPuntosInsigniasGrupo

Method: GET

URL:

http://ssii2014.e-ucm.es:80//OlimpiadaFDIServices/rest/insignias/puntosInsigniasGrupo/{idGrupo}

{idGrupo} ejemplo = 2

Este servicio recibe por la url directamente el id del grupo del cual queremos conocer sus puntos, se encarga de comprobar que el grupo exista y en caso de que sea así devuelve la suma de todos los puntos conseguidos por los usuarios pertenecientes al grupo

Response:


```
{
  "code": 0,
  "message": "OK",
  "result": 130
}
```

## 9) RestService getPuntosInsigniasUsuario

Method: GET

URL:

http://ssii2014.e-ucm.es:80//OlimpiadaFDIServices/rest/insignias/puntosInsigniasUsuario/{usuario}

{usuario} ejemplo = alejandro

Este servicio recibe por la url directamente el nombre del usuario del cual queremos conocer sus puntos, se encarga de comprobar que el usuario exista y en caso de que sea así devuelve la suma de todos los puntos de sus insignias

Response:


```
{
  "code": 0,
  "message": "OK",
  "result": 65
}
```

## 10) RestService getGrupoPorUsuario

Method: GET

URL:

http://ssii2014.e-ucm.es:80/OlimpiadaFDIServices/rest/insignias/grupoPorUser/{nombreUsuario}

{nombreUsuario} ejemplo = User1

Este servicio recibe por la url directamente el nombre del usuario del cual queremos conocer su grupo, Devuelve el id del grupo al que pertenece

Result:

```
{
  "code":0,
  "message":"OK",
  "result":2
}
```

## 11) RestService getUsuariosDeGrupo

Method: GET

URL:

http://ssii2014.e-ucm.es:80/OlimpiadaFDIServices/rest/insignias/usuariosDeGrupo/{idGrupo}

{idGrupo} ejemplo = 2

Este servicio recibe por la url directamente el id del grupo del cual queremos conocer sus usuarios, Devuelve una lista con los nombres de los usuarios que pertenecen a ese grupo

Result:


```
{
        "code":0,
        "message":"OK",
        "result":["Miguel","User1"]
}
```

## 12) RestService obtenerTodosGrupo

Method: GET

URL:

http://ssii2014.e-ucm.es:80/OlimpiadaFDIServices/rest/insignias/obtenerTodosGrupos

Este servicio no recibe parámetros y devuelve todos los grupos existentes

Result:


```
{
        "code":0,
        "message":"OK",
        "result":[{"idGrupo":1,"descripcion":"Grupo one"},
                {"idGrupo":2,"descripcion":"Grupo
        two"},
                {"idGrupo":289,"descripcion":"The
              manimals"}
              ]
}
```

## 13) RestService obtenerTodosUsuarios

Method: GET

URL:

http://ssii2014.e-ucm.es:80/OlimpiadaFDIServices/rest/insignias/obtenerTodosUsuarios

Este servicio no recibe parámetros y devuelve todos los usuarios existentes

Result:


```
{
        "code":0,
        "message":"OK",
        "result":[
                {"idUsuario":1419951352244,"usuario":"User1","correo":"user1@ucm.es"},
                {"idUsuario":1421011435564,"usuario":"Miguel","correo":"miguel@olimpiada.fdi.ucm.es"}
                ]
}
```